

Frontiers
in
Artificial
Intelligence
and
Applications

SOFT COMPUTING SYSTEMS

Design, Management and Applications

Edited by
Ajith Abraham
Javier Ruiz-del-Solar
Mario Köppen

IOS
Press

Ohmsha

VISIT...

LANZAROTE
Caliente.COM

SOFT COMPUTING SYSTEMS

Frontiers in Artificial Intelligence and Applications

Volume 87

Published in the subseries

Knowledge-Based Intelligent Engineering Systems

Editors: L.C. Jain and R.J. Howlett

Recently published in KBIES:

- Vol. 86, In production
- Vol. 83, In production
- Vol. 82, E. Damiani et al. (Eds.), Knowledge-Based Intelligent Information Engineering Systems and Allied Technologies
- Vol. 79, H. Motoda (Ed.), Active Mining
- Vol. 72, A. Namatame et al. (Eds.), Agent-Based Approaches in Economic and Social Complex Systems
- Vol. 69, N. Baba et al. (Eds.), Knowledge-Based Intelligent Information Engineering Systems & Allied Technologies

Recently published in FAIA:

- Vol. 85, J.M. Abe and J.I. da Silva Filho (Eds.), Advances in Logic, Artificial Intelligence and Robotics
- Vol. 84, H. Fujita and P. Johannesson (Eds.), New Trends in Software Methodologies, Tools and Techniques
- Vol. 81, In production
- Vol. 80, T. Welzer et al. (Eds.), Knowledge-based Software Engineering
- Vol. 78, T. Vidal and P. Liberatore (Eds.), STAIRS 2002
- Vol. 77, F. van Harmelen (Ed.), ECAI 2002
- Vol. 76, P. Šinčák et al. (Eds.), Intelligent Technologies – Theory and Applications
- Vol. 75, I.F. Cruz et al. (Eds.), The Emerging Semantic Web
- Vol. 74, M. Blay-Fornarino et al. (Eds.), Cooperative Systems Design
- Vol. 73, H. Kangassalo et al. (Eds.), Information Modelling and Knowledge Bases XIII
- Vol. 71, J.M. Abe and J.I. da Silva Filho (Eds.), Logic, Artificial Intelligence and Robotics
- Vol. 70, B. Verheij et al. (Eds.), Legal Knowledge and Information Systems
- Vol. 68, J.D. Moore et al. (Eds.), Artificial Intelligence in Education
- Vol. 67, H. Jaakkola et al. (Eds.), Information Modelling and Knowledge Bases XII
- Vol. 66, H.H. Lund et al. (Eds.), Seventh Scandinavian Conference on Artificial Intelligence
- Vol. 65, In production
- Vol. 64, J. Breuker et al. (Eds.), Legal Knowledge and Information Systems
- Vol. 63, I. Gent et al. (Eds.), SAT2000
- Vol. 62, T. Hruška and M. Hashimoto (Eds.), Knowledge-Based Software Engineering
- Vol. 61, E. Kawaguchi et al. (Eds.), Information Modelling and Knowledge Bases XI
- Vol. 60, P. Hoffman and D. Lemke (Eds.), Teaching and Learning in a Network World
- Vol. 59, M. Mohammadian (Ed.), Advances in Intelligent Systems: Theory and Applications
- Vol. 58, R. Dieng et al. (Eds.), Designing Cooperative Systems
- Vol. 57, M. Mohammadian (Ed.), New Frontiers in Computational Intelligence and its Applications
- Vol. 56, M.I. Torres and A. Sanfeliu (Eds.), Pattern Recognition and Applications
- Vol. 55, G. Cumming et al. (Eds.), Advanced Research in Computers and Communications in Education

ISSN: 0922-6389

Soft Computing Systems

Design, Management and Applications

Edited by

Ajith Abraham

*Computer Science Department, Oklahoma State University,
Tulsa, USA*

Javier Ruiz-del-Solar

*Department of Electrical Engineering, Universidad de Chile,
Santiago, Chile*

and

Mario Köppen

Fraunhofer IPK, Berlin, Germany



Amsterdam • Berlin • Oxford • Tokyo • Washington, DC

© 2002, The authors mentioned in the Table of Contents

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, without prior written permission from the publisher.

ISBN 1 58603 297 6 (IOS Press)

ISBN 4 274 90558 6 C3055 (Ohmsha)

Library of Congress Control Number: 2002113704

Publisher

IOS Press

Nieuwe Hemweg 6B

1013 BG Amsterdam

The Netherlands

fax: +31 20 620 3419

e-mail: order@iospress.nl

Distributor in the UK and Ireland

IOS Press/Lavis Marketing

73 Lime Walk

Headington

Oxford OX3 7AD

England

fax: +44 1865 75 0079

Distributor in the USA and Canada

IOS Press, Inc.

5795-G Burke Centre Parkway

Burke, VA 22015

USA

fax: +1 703 323 3668

e-mail: iosbooks@iospress.com

Distributor in Germany, Austria and Switzerland

IOS Press/LSL.de

Gerichtsweg 28

D-04103 Leipzig

Germany

fax: +49 341 995 4255

Distributor in Japan

Ohmsha, Ltd.

3-1 Kanda Nishiki-cho

Chiyoda-ku, Tokyo 101-8460

Japan

fax: +81 3 3233 2426

LEGAL NOTICE

The publisher is not responsible for the use which might be made of the following information.

PRINTED IN THE NETHERLANDS

Preface

Hybridization of intelligent systems is a promising research field of modern artificial/computational intelligence concerned with the development of the next generation of intelligent systems. A fundamental stimulus to the investigations of Hybrid Intelligent Systems (HIS) is the awareness in the academic communities that combined approaches will be necessary if the remaining tough problems in artificial/computational intelligence are to be solved. Recently, hybrid intelligent systems are getting popular due to their capabilities in handling several real world complexities involving imprecision, uncertainty and vagueness. Current research interests in this field focus on integration of the different soft computing paradigms like fuzzy logic, neurocomputation, evolutionary computation, probabilistic computing and their interactions with hard computing techniques, intelligent agents, machine learning, other intelligent computing frameworks and so on. The phenomenal growth of hybrid intelligent systems and related topics has created the need for this International Conference as a venue to present the latest research. HIS'02 builds on the success of last year's. HIS'01 was held in Adelaide, Australia, 11–12 December 2001 and attracted participants from over 26 countries.

HIS'02, the Second International Conference on Hybrid Intelligent Systems, took place in Santiago de Chile, December 1–4, 2002, addressing the following four important themes:

- Hybrid soft computing
- Soft computing for pattern recognition and signal processing
- Soft computing for multimedia and Internet modeling
- Intelligent data mining

HIS'02 was hosted by the Faculty of Physical and Mathematical Sciences at the Universidad de Chile. HIS'02 is technically co-sponsored by The World Federation on Soft Computing, ENNS (European Neural Network Society), EUSFLAT (European Society for Fuzzy Logic and Technology), EVONET (European Network of Excellence in Evolutionary Computing), IEEE Region 9 and IOS Press. HIS'02 program committee represented 21 countries on 5 continents and authors submitted 132 papers from 38 countries on 5 continents. This certainly attests to the wide-spread, international importance of the theme of the conference. Each paper was peer reviewed by at least two independent referees of the program committee and based on the recommendation of the reviewers 87 papers were finally accepted. HIS'02 also received 9 technical sessions and 10 tutorial proposals addressing different aspects of intelligent systems and applications.

HIS'02 was blessed to have the following plenary speakers:

1. Yasuhiko Dote, '*Neuro-fuzzy control*'
2. Janusz Kacprzyk, '*Protoforms of linguistic data summaries: towards more general natural-language-based data mining tools*'
3. Erkki Oja, '*Independent component analysis*'
4. Juan Carlos Letelier, '*Anticipatory computing with autopoietic and (M,R)-Systems*'

5. Bernard De Baets, '*Fuzzy set theory: a playground for mathematicians*'
6. Oussama Khatib, '*Robots for the human and haptic interaction*'
7. William Langdon, '*A hybrid genetic programming neural network classifier for use in drug discovery*'
8. Andrew Sung, '*Role of soft computing in internet security*'

We would like to express our sincere gratitude to all the authors and members of the program and local organizing committees that has made this conference a success. We are also grateful to The Center for Web Research, Chile and Aglaia GmbH, Germany for generously sponsoring the travel grant funds. Our special thanks also to the plenary speakers and tutorial presenters for their effort in preparing the lectures. We are also indebted to Lakhmi Jain (Editor of Knowledge-Based Intelligent Engineering Systems Series, IOS Press, Netherlands) for the timely advices related to this volume.

The editors would like to thank E.H. Fredriksson, Carry Koolbergen and Anne Marie de Rover of IOS Press, Netherlands for the support and excellent cooperative collaboration to produce this important scientific work. Last, but not the least, we would like to express our gratitude to our colleagues from the Department of Computer Science at Oklahoma State University, USA, Department of Pattern Recognition at Fraunhofer IPK-Berlin, Germany and Department of Electric Engineering at Universidad de Chile who assisted us in the organization of HIS'02.

September 2002

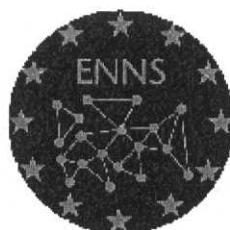
Ajith Abraham, Mario Köppen and Javier Ruiz-del-Solar
Editors

Field Editors

Aureli Soria-Frisch, Fraunhofer IPK-Berlin, Germany
 Fabio Abbatista, Universitat di Bari, Italy
 Raul Vicente-Garcia, Fraunhofer IPK-Berlin, Germany

HIS 2002
 Hybrid Intelligent Systems

HIS'02 Technical Co-Sponsors



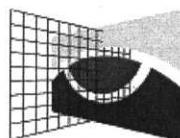
European Neural Network Society



European Network of Excellence in Evolutionary Computing



HIS02 Fellowship Sponsors



Aglaia
Gesellschaft für Bildverarbeitung
und Kommunikation mbH

HIS'02 – Organization

Honorary Chairman

Yasuhiko Dote, Muroran Institute of Technology, Japan

General Chairmen

Ajith Abraham, Oklahoma State University, USA, Email: <ajith.abraham@ieee.org>

Mario Köppen, Fraunhofer IPK-Berlin, Germany, Email: <mario.koeppen@ipk.fhg.de>

Javier Ruiz-del-Solar, Universidad de Chile, Chile, Email: <jruizd@cec.uchile.cl>

Local Organizing Chair

Javier Ruiz-del-Solar, Universidad de Chile, Chile

Area Chairs

Track1: Hybrid Soft Computing

Ajith Abraham, Oklahoma State University, USA

Track 2: Soft Computing for Pattern Recognition and Signal Processing

Mario Köppen, Fraunhofer IPK-Berlin, Germany

Track 3: Soft Computing and the Internet

Fabio Abbattista, Università di Bari, Italy

Track 4: Intelligent Data Mining

Kate Smith, Monash University, Australia

Special Sessions Chair

Xiao-Zhi Gao, Helsinki University of Technology, Finland

Travel Grants Chair

Katrin Franke, Fraunhofer IPK-Berlin, Germany

Web Chairs

Mauricio Gaete, Universidad de Chile, Chile

Diego Sepúlveda, Universidad de Chile, Chile

Finance Chair

Roberto Avilés, Universidad de Chile, Chile

Local Organizing Committee

Roberto Avilés, Universidad de Chile, Chile

Rodrigo Palma, Universidad de Chile, Chile

Richard Weber, Universidad de Chile, Chile

International Technical Committee

Sung-Bae Cho, Yonsei University, Korea
 Kate Smith, Monash University, Australia
 Hung T. Nguyen, New Mexico State University, USA
 Tom Gedeon, Murdoch University, Australia
 Robert John, De Montfort University, UK
 Fabio Abbatista, Universitat di Bari, Italy
 Bernard de Baets, Ghent University, Belgium
 Carlos A. Coello Coello, Laboratorio Nacional de Informática Avanzada, Mexico
 Vladimir Kvasnicka, Slovak Technical University, Slovakia
 Zensho Nakao, University of Ryukyus, Japan
 Witold Pedrycz, University of Alberta, Canada
 Udo Seiffert, University of South Australia, Australia
 Jarno Tanskanen, University of Kuopio, Finland
 Hamid R. Tizhoosh, University of Waterloo, Canada
 Olgierd Unold, Wroclaw University of Technology, Poland
 Vasant Honavar, Iowa State University, USA
 Maumita Bhattacharya, Monash University, Australia
 José Manuel Benítez, University of Granada, Spain
 Janusz Kacprzyk, Polish Academy of Sciences, Poland
 Sankar K Pal, Indian Statistical Institute, India
 Vijayan Asari, Old Dominion University, USA
 Dharmendhra Sharma, University of Canberra, Australia
 Morshed U Chowdhury, Deakin University, Australia
 Xiao Zhi Gao, Helsinki University of Technology, Finland
 Costa Branco P J, Instituto Superior Technico, Portugal
 Zorica Nedic, University of South Australia, Australia
 Rajkumar Roy, Cranfield University, United Kingdom
 Katrin Franke, Fraunhofer IPK-Berlin, Germany
 Aureli Soria-Frisch, Fraunhofer IPK-Berlin, Germany
 Xiufen Liu, Fraunhofer IPK-Berlin, Germany
 Nikhil R. Pal, Indian Statistical Institute, India
 Ricardo Baeza-Yates, Universidad de Chile, Chile
 Kaori Yoshida, Kyushu Institute of Technology, Japan
 William B. Langdon, University College London, UK
 Akira Asano, Hiroshima University, Japan
 Evgenia Dimitriadou, Vienna University of Technology, Austria
 Francisco Herrera, University of Granada, Spain
 Stefano Cagnoni, University of Parma, Italy
 Marley Vellasco, PUC-RJ, Brazil
 André Ferreira de Carvalho, Universidade de São Paulo, Brazil
 Antônio de Pádua Braga, Federal University of Minas Gerais, Brazil
 Etienne Kerre, Ghent University, Belgium
 Frank Hoffmann, Royal Institute of Technology, Sweden
 Günther Raidl, Vienna University of Technology, Austria
 Greg Huang, MIT, USA
 Sami Khuri, San Jose University, USA
 Hans Heinrich Bothe, Technical University of Denmark, Denmark
 Andreas König, Technical University of Dresden, Germany
 Vladislav B. Valkovsky, Uppsala University, Sweden

Luis Magdalena, Universidad Politecnica de Madrid, Spain
L.C. Jain, University of South Australia, Australia
Jose Ramon Alvarez Sanchez, Univ. Nac.de Educacion a Distancia, Spain
Janos Abonyi, University of Veszprem, Hungary
Hans-Michael Voigt, GFAI, Germany
Jose Mira, UNED, Spain
Andrew Sung, New Mexico Institute of Mining and Technology, USA
Rajiv Khosla, La Trobe University, Australia

Additional Referees

Daniel Kottow
Mike Nachtegaele
Dietrich Vanderweken
Christian Veenhuis
Ottmar Bünnenmeyer

Contents

Preface	v
HIS'02 Technical Co-Sponsors	vii
HIS'02 Organization	viii
1. Abstracts of the Plenary Presentations	1
2. Neural Systems	
The Use of a Back Propagation Neural Network to Determine the Load Distribution on a Component, <i>R. Amali, J. Vinney, S. Noroozi and V. Patel</i>	15
Performance-guided Neural Network for Rapidly Self-organising Active Network Management, <i>S.W. Lee, D. Palmer-Brown, J. Tepper and C. Roadknight</i>	21
An Automated Hybrid Reasoning System for Forecasting, <i>F. Fdez-Riverola and J.M. Corchado</i>	32
Rule Extraction from Bagged Neural Networks, <i>G. Bologna</i>	42
Nonlinear Principal Component Analysis to Preserve the Order of Principal Components, <i>R. Saegusa and S. Hashimoto</i>	54
A Neural Network Model of Rule-guided Behavior, <i>T. Minami and T. Inui</i>	64
Selection of Models for Time Series Prediction via Meta-Learning, <i>R.B. Cavalcante Prudêncio and T.B. Ludermir</i>	74
Spiking Neurons in Clustering of Diabetic Retinopathy Data, <i>K.J. Cios, W. Jackson, W. Swiercz and L. Springhetti</i>	84
3. Fuzzy Sets	
A Fuzzy Relatedness Measure for Determining Interestingness of Association Rules, <i>B. Shekar and R. Natarajan</i>	95
Factor Analysis with Qualitative Factors as Fuzzy Numbers, <i>E. Rakus-Andersson and L. Zakrzewski</i>	105
An Imperfect String Matching Experience using Deformed Fuzzy Automata, <i>J.J. Astrain, J.R. Garitagoitia, J. Villadangos, F. Fariña and A. Córdoba</i>	115
An XML-based Specification of Fuzzy Logic Controllers, <i>D. Mastropasqua, N. Mosca and F. Zambetta</i>	124
Comparison of Fuzzy Rule Selection Criteria for Classification Problems, <i>H. Ishibuchi and T. Yamamoto</i>	132
Linguistic Hedges: A Quantifier Based Approach, <i>M. De Cock</i>	142
4. Evolutionary Computation	
Analyzing the Founder Effect in Simulated Evolutionary Processes using Gene Expression Programming, <i>C. Ferreira</i>	153
Hybrid Evolutionary Multi-Objective Optimization Algorithms, <i>H. Ishibuchi and T. Yoshida</i>	163
A Permutation Based Genetic Algorithm for RNA Secondary Structure Prediction, <i>K.C. Wiese and E. Glen</i>	173
Design of Strong Causal Fitness Functions, <i>S. Hirche, I. Santibanez-Koref and I. Boblan</i>	183
Noise and Elitism in Evolutionary Computation, <i>T. Beker and L. Hadany</i>	193

5. Machine Learning

Anticipatory Computing with Autopoietic and (M,R) Systems, <i>J.C. Letelier, G. Marín, J. Mpodozis and J. Soto-Andrade</i>	205
An Efficient Algorithm in Optimal Partition Problem for Trees Induction, <i>M. Asseraf</i>	212
Condition Monitoring, Root Cause Analysis and Decision Support on Urgency of Actions, <i>G. Weidl, A. Madsen and E. Dahlquist</i>	221
Towards a Unique Framework to Describe and Compare Diagnosis Approaches, <i>C.H. Zanni, M. Le Goc and C.S. Frydman</i>	231
Experimental Evaluation of the PLA-based Permutation-Scheduling, <i>J. Jedrzejowicz and P. Jedrzejowicz</i>	241
A Study of K -Nearest Neighbour as an Imputation Method, <i>G.E.A.P.A. Batista and M.C. Monard</i>	251
Determining the Degree of Generalization using an Incremental Learning Algorithm, <i>P. Zegers and M.K. Sundareshan</i>	261
Towards Landscape Analyses to Inform the Design of Hybrid Local Search for the Multiobjective Quadratic Assignment Problem, <i>J.D. Knowles and D.W. Corne</i>	271
Active Learning using One-class Classification, <i>I. Gokcen, J. Peng and B.P. Buckles</i>	280
Enhancing Real-World Applicability by Providing Confidence-in-Prediction in the XCS Classifier System, <i>P.W. Dixon, D.W. Corne and M.J. Oates</i>	290
Latent Semantic Indexing Based on Factor Analysis, <i>N. Kawamae</i>	300
Document Oriented Modeling of Cellular Automata, <i>C. Veenhuis and M. Köppen</i>	309

6. Support Vector Machines and Kernel Methods

An Empirical Comparison of Kernel Selection for Support Vector Machines, <i>A.B.M. Shawkat Ali and A. Abraham</i>	321
Adaptive Support Vector Classifications, <i>Z. Liu and Y. Xu</i>	331
Mercer's Kernel Based Learning for Fault Detection, <i>B. Ribeiro and P. Carvalho</i>	341
Performance Based Feature Identification for Intrusion Detection using Support Vector Machines, <i>S. Mukkamala and A.H. Sung</i>	351

7. Features and Classification

A Trainable Classifier via k Nearest Neighbors, <i>I. Mora-Jiménez, A. Lyhyaoui, J. Arenas-García, A. Navia-Vázquez and A.R. Figueiras-Vidal</i>	365
Combining Classifiers with Multimethod Approach, <i>M. Lenič and P. Kokol</i>	374
Feature Extraction by Distance Neural Network in Classification Tasks, <i>C. Maturana and R. Weber</i>	384
Revealing Feature Interactions in Classification Tasks, <i>D. Partridge and S. Cang</i>	394
Ensembles in Practice: Predication, Estimation, Multi-Feature and Noisy Data, <i>S. Zemke</i>	404

8. Data Mining

Protoforms of Linguistic Data Summaries: Towards More General Natural-Language-based Data Mining Tools, <i>J. Kacprzyk and S. Zadrozny</i>	417
Sparse Distributed Memory with Adaptive Threshold, <i>J.L. Aguilar and N. Perozo</i>	426
UniLR: An Automated Fuzzy Legal Reasoner, <i>D. Sharma</i>	433
Set Approximation Quality Measures in the Variable Precision Rough Set Model, <i>W. Ziarko</i>	442
A Data Preparation Bayesian Approach for a Clustering Genetic Algorithm, <i>E.R. Hruschka Jr., E.R. Hruschka and N.F.F. Ebecken</i>	453
Reconstruction of Conditional Distribution Field based on Empirical Data, <i>C.A. Jakovlevich</i>	462

Bi-directional Flow of Information in the Softboard Architecture, <i>S. Macedo and E. Mamdani</i>	470
Voice Codification using Self Organizing Maps as Data Mining Tool, <i>J.D. Velásquez, H. Yasuda, T. Aoki and R. Weber</i>	480
Identifying Patterns of Corporate Tax Payment, <i>M.C. Martins and I.M. Garaffa</i>	490
Self-organized Data and Image Retrieval as a Consequence of Inter-dynamic Synergistic Relationships in Artificial Ant Colonies, <i>V. Ramos, F. Muge and P. Pina</i>	500
9. Agent Architectures	
Designing Not-So-Dull Virtual Dolls, <i>F. Zambetta and G. Catucci</i>	513
SADISCO: A Scalable Agent Discovery and Composition Mechanism, <i>J.J. Nolan, A.K. Sood and R. Simon</i>	519
MAYBE - Multi-Agent Yield-Based Engineering : Improve Training in the Emergency Room Chain, <i>S. Gouarderes, G. Gouarderes and P. Delpy</i>	529
3D-CG Avatar Motion Design by Means of Interactive Evolutionary Computation, <i>H. Iba, N. Tokui and H. Wakaki</i>	540
Alliance Formation with Several Coordinators, <i>V. Marik and V. Mashkov</i>	550
10. Web Computing	
Balancing Volume, Quality and Freshness in Web Crawling, <i>R. Baeza-Yates and C. Castillo</i>	565
Learnable Topic-specific Web Crawler, <i>N. Angkawattanawit and A. Rungsawang</i>	573
A Spatial Dimension for Searching the World Wide Web, <i>M.A. Rodríguez Tastets</i>	583
Building Yearbooks with RDF, <i>E.K. Morales and C. Gutiérrez</i>	593
A Non-Deterministic versus Deterministic Algorithm for Searching Spatial Configurations, <i>M.C. Jarur and M.A. Rodríguez</i>	602
Parallel Text Query Processing using Composite Inverted Lists, <i>M. Marín</i>	612
11. Interactive Systems	
Human Reasoning in Soft Computing, <i>V.S. Kumar</i>	625
A Focus and Constraint-based Genetic Algorithm for Interactive Directed Graph Drawing, <i>H.A.D. Do Nascimento and P. Eades</i>	634
Dialogue Act Connectionist Detection in a Spoken Dialogue System, <i>E. Sanchis and M.J. Castro</i>	644
A Trial Method to Create a Natural Interaction in Interactive Genetic Algorithm, <i>F. Sugimoto and M. Yoneyama</i>	652
12. Signal and Image Processing	
Eigenspace-based Face Recognition: A Comparative Study of Different Hybrid Approaches, <i>P. Navarrete and J. Ruiz-Del-Solar</i>	663
Pattern Recognition with Ultrasonic Sensor using Classification Methods, <i>N. Ait Oufroukh and E. Colle</i>	673
A Color Pattern Recognition Problem based on the Multiple Classes Random Neural Network Model, <i>J. Aguilar</i>	681
Jigsawing : A Method to Create Virtual Examples in OCR Data, <i>S.V.N. Vishwanthan and M.N. Murty</i>	690
Model-based Restoration of Short-Exposure Solar Images, <i>M. Haindl and S. Šimberová</i>	697
Using a Helper FFN to Represent the Cost Function for Training DRNN's by Gradient Descent, <i>R.K. Brouwer</i>	707

Scene-based Nonuniformity Correction Method using the Inverse Covariance Form of the Kalman Filter, <i>S. Torres and J. Pezoa</i>	715
Adaptive Bias Compensation for Non-Uniformity Correction on Infrared Focal Plane Array Detectors, <i>E. Vera, R. Reeves and S. Torres</i>	725
Combining Genetic Algorithms and Neural Networks to Build a Signal Pattern Classifier, <i>R.S. Youssif and C.N. Purdy</i>	735
Accurate Human Face Extraction using Genetic Algorithm and Subspace Method, <i>M. Murakami, M. Yoneyama and K. Shirai</i>	745
The Evolutionary Learning Rule for System Identification in Adaptive Finite Impulse Filters, <i>O. Montiel, O. Castillo, P. Melin and R. Sepulveda</i>	755
2D-Histogram Lookup for Low-Contrast Fault Processing, <i>M. Köppen, R. Vicente Garcia, X. Liu and B. Nickolay</i>	765
13. Industrial Applications	
A Railway Interlocking Safety Verification System based on Abductive Paraconsistent Logic Programming, <i>K. Nakamatsu, J.M. Abe and A. Suzuki</i>	775
Complete Algorithm to Realize CI Model-based Control and Monitoring Strategies on Microcontroller Systems, <i>K.-D. Kramer, S. Patzwahl and T. Nacke</i>	785
Using Genetic Algorithms for Minimizing the Production Costs of Hollow Core Slabs, <i>V.C. Castilho, M.C. Nicoletti and M.K. El Debs</i>	796
Recognizing Malicious Intention in an Intrusion Detection Process, <i>F. Cuppens, F. Autrel, A. Miège and S. Benferhat</i>	806
A Self-Growing Probabilistic Decision-based Neural Network with Applications to Anchor/Speaker Identification, <i>S.S. Cheng, Y.H. Chen, C.L. Tseng, H.C. Fu and H.T. Pao</i>	818
HyCAR - A Robust Hybrid Control Architecture for Autonomous Robots, <i>F.J. Heinen and F.S. Osório</i>	830
14. Web and Multimedia Applications	
Clustering Web User Interests using Self Organising Maps, <i>X. Wang and K.A. Smith</i>	843
Web Traffic Mining using a Concurrent Neuro-Fuzzy Approach, <i>X. Wang, A. Abraham and K.A. Smith</i>	853
Panoramic View System for Extracting Key Sentences based on Viewpoints and an Application to a Search Engine, <i>W. Sunayama and M. Yachida</i>	863
Frequent Flyer Points Calculator: More than just a Table Lookup, <i>G.W. Rumanthir</i>	871
A Large Benchmark Dataset for Web Document Clustering, <i>M.P. Sinka and D.W. Corne</i>	881
Simulating an Information Ecosystem within the WWW, <i>R.L. Walker</i>	891
Author Index	901

Section 1

Abstracts of the Plenary Presentations

This page intentionally left blank

HIS02 Plenary Presentations

Independent Component Analysis

Erkki Oja

*Helsinki University of Technology,
Neural Networks Research Centre,
P.O.B. 5400, 02015 HUT, Finland
Erkki.Oja@hut.fi*

Abstract. Independent Component Analysis (ICA) is a computational technique for revealing hidden factors that underlie sets of measurements or signals. ICA assumes a statistical model whereby the observed multivariate data, typically given as a large database of samples, are assumed to be linear or nonlinear mixtures of some unknown latent variables. The mixing coefficients are also unknown. The latent variables are non gaussian and mutually independent, and they are called the independent components of the observed data. By ICA, these independent components, also called sources or factors, can be found. Thus ICA can be seen as an extension to Principal Component Analysis and Factor Analysis. ICA is a much richer technique, however, capable of finding the sources when these classical methods fail completely.

In many cases, the measurements are given as a set of parallel signals or time series. Typical examples are mixtures of simultaneous sounds or human voices that have been picked up by several microphones, brain signal measurements from multiple EEG sensors, several radio signals arriving at a portable phone, or multiple parallel time series obtained from some industrial process. The term blind source separation is used to characterize this problem.

The lecture will first cover the basic idea of demixing in the case of a linear mixing model and then take a look at the recent nonlinear demixing approaches.

Although ICA was originally developed for digital signal processing applications, it has recently been found that it may be a powerful tool for analyzing text document data as well, if the documents are presented in a suitable numerical form. A case study on analyzing dynamically evolving text is covered in the talk.

Fuzzy Set Theory: a Playground for Mathematicians

Bernard De Baets

Department of Applied Mathematics, Biometrics and Process Control

Ghent University, Coupure links 653, 9000 Gent, BELGIUM

Bernard.DeBaets@rug.ac.be

Abstract. When Zadeh first introduced the notion of a fuzzy set, it was laughed away by the ruling caste of 'serious' mathematicians and logicians. Driven by ignorance, the main reason for it was probably strategic: as fuzzy set theory touches upon the very building block of mathematics, it might be better to deny it than to risk being taken out of business. Even the ample practical realizations had no effect on them.

Unfortunately, the fuzzy mathematics community has offered plenty of stones to be thrown at them. One by one, the core mathematical theories were subjected to fuzzification: topology, algebra, analysis, etc. The literature has witnessed various waves of these activities, not in the least the 'replace min by a triangular norm' wave. The majority of these works are sheer mathematical exercises and have no reason to appeal to the established mathematical community.

Indeed, instead of trying to generalize things on a micro-level, the potential interest of fuzzy set theory lies on the macro-level, offering a global view and additional insight into existing mathematical theories. Fortunately, more and more works of this kind are appearing, and are, finally, attracting the interest of non-fuzzy mathematicians. A nice work of the latter kind is the far-reaching monograph "Metamathematics of fuzzy logic" by Hajek on fuzzy logic in 'narrow sense', making its way into the established community of many-valued logicians. Another example is the book "Triangular norms" of Klement, Mesiar and Pap, dealing with the omnipresent concept of a triangular norm, originating from the theory of probabilistic metric spaces, but elegantly further explored by this trio of fine 'fuzzy' mathematicians.

The purpose of this lecture is not to bore the listeners with a historic account of the contributions that fuzzy set theorists did make to the development of mathematics, but to offer the audience a digestible selection of mathematical appetizers illustrating what fuzzy set theory has to offer. One thing is clear: it takes a multi-skilled mathematician to succeed. Without the purpose of being pretentious, the author will discuss a number of developments he was privileged to be involved in. Possible topics include uninorms (ranging from semi-group aspects to applications in expert systems), fuzzy preference structures (ranging from functional equations to applications in decision support systems) and similarity measures (ranging from inequalities in quantum-mechanics to applications in numerical taxonomy).

Robots for the Human and Haptic Interaction

Oussama Khatib

Department of Computer Science

Stanford University

khatib@cs.stanford.edu

Abstract. A new field of robotics is emerging. Robots are today moving towards applications beyond the structured environment of a manufacturing plant. They are making their way into the everyday world that people inhabit. The successful introduction of robotics into human environments will rely on the development of competent and practical systems that are dependable, safe, and easy to use. The discussion focuses on models, strategies, and algorithms associated with the autonomous behaviours needed for robots to work, assist, and cooperate with humans. In addition to the new capabilities they bring to the physical robot, these models and algorithms and more generally the body of developments in robotics is having a significant impact on the virtual world. Haptic interaction with an accurate dynamic simulation provides unique insights into the real-world behaviors of physical systems. The potential applications of this emerging technology include virtual prototyping, animation, surgery, robotics, cooperative design, and education among many others. Haptics is one area where the computational requirement associated with the resolution in real-time of the dynamics and contact forces of the virtual environment is particularly challenging. The presentation describes various methodologies and algorithms that address the computational challenges associated with interactive simulations involving multiple contacts and impacts with complex human-like structures.

A Hybrid Genetic Programming Neural Network Classifier for Use in Drug Discovery

William B. Langdon

Computer Science

University College, London,

London, WC1E 6BT, UK

W.B.Langdon@cs.ucl.ac.uk

Abstract. We have shown genetic programming (GP) can automatically fuse given classifiers of diverse types to produce a hybrid classifier. Combinations of neural networks, decision trees and Bayes classifier have been formed.

On a range of benchmarks the evolved multiple classifier system is better than all of its components. Indeed its Receiver Operating Characteristics (ROC) are better than [Scott et al., 1998]'s "Maximum Realisable Receiver Operating Characteristics" MRROC (convex hull)

An important component in the drug discovery is testing potential drugs for activity with P450 cell membrane molecules. Our technique has been used in a blind trial where artificial neural networks are trained by Clementine on P450 pharmaceutical data. Using just the trained networks, GP automatically evolves a composite classifier. Recent experiments with boosting the networks will be compared with genetic programming.

Protoforms of Linguistic Data Summaries: Towards More General Natural-Language-Based Data Mining Tools

Janusz Kacprzyk, Sawomir Zadrony
*Systems Research Institute, Polish Academy of Sciences,
ul. Newelska 6, 01-447 Warsaw, Poland
{kacprzyk, zadrozny}@ibspan.waw.pl*

Abstract. We advocate the use of linguistic data summaries, exemplified by "most employees are young and well paid", for a personnel database, as an intuitive and human consistent tool for data mining. We present an interactive approach, based on fuzzy logic and fuzzy database queries, which makes it possible to implement such summaries. We show how fuzzy queries are related to linguistic summaries, and show that one can introduce a hierarchy of prototype forms (proto forms), or abstract summaries in the sense of latest Zadeh's [1] ideas meant mainly for increasing deduction capabilities of search engines. For illustration we show an implementation for a sales database in a computer retailer, employing some type of a proto form of a linguistic summary. (see also paper in this book)

Role of Soft Computing in Internet Security

Andrew H. Sung

Department of Computer Science &

Institute for Complex Additive Systems Analysis

New Mexico Tech

Socorro, NM 87801, U.S.A.

Abstract. Information security (INFOSEC) is an issue of global concern. As the Internet is bringing great convenience and benefits to the modern society, the rapidly increasing connectivity and accessibility to the Internet is also posing a serious threat to security and privacy, to individuals, organizations, and nations alike. Finding effective ways to detect, prevent, and respond to intrusions and hacker attacks of networked computers and information systems, therefore, has become a timely research topic.

In this talk, I will present the application of soft computing methods in information security. Since intrusions and malicious attacks are commonly detected by inspecting the system audit data, soft computing methods can be utilized in various ways for analyzing, organizing, and mining the tremendous amount of audit data generated by computer systems.

I will also describe several of our ongoing INFOSEC research projects that use soft computing methods: intrusion detection using Support Vector Machines(SVMs); feature ranking and selection for intrusion detection; and steganalysis (stegonagraphy detection). Our research demonstrates the soft computing paradigms tremendous potential for problem solving in information security, e.g., SVMs are ideal learning machines for intrusion detection, and neural networks and fuzzy systems have various applications in INFOSEC problems as well.

Neuro-Fuzzy Control

Yasuhiko Dote

Muroran Institute of Technology, Japan

dote@epsilon2.csse.muroran-it.ac.jp

Abstract. In 1987, fuzzy control was successfully applied in industrial plants in Japan. In the late eighties, neuro-control was used for robot arms including the robot arm of the space shuttle, chemical processes, continuous production of high- quality parts, and aerospace applications) in the U.S.A.. In 1991, the BISC (Berkley Initiative in Soft Computing) was established as an ILP (industrial liaison program), with Dr. Zadeh, as its director. Since the establishment of BISC, researchers through the world have been studying soft computing, i.e., the fusion of fuzzy logic (FL), neural networks (NN), and evolutionary computation (EC). The term computational intelligence as defined by Dr. L. A. Zadeh, is the combination of soft computing and numerical processing. First this term was used in 1990 by the IEEE Neural Networks Council. Three IEEE International Workshop on Soft Computing in Industry has been held in Muroran, Japan in 1993, 1996, and 1999, with Dr. Zadeh as plenary speaker each time . The first workshop put emphasis on the fusion of neural networks and fuzzy logic. In the second workshop, evolutionary computation ,chaotic computing, and immune networks were discussed. The third workshop focused on cognitive distributed artificial intelligence (human-like information processing and reactive distributed artificial intelligence (bioinformatic information processing) . The IEEE international workshops on applications of industries were held in Finland and the U.S.A. in 1999 and 2001, respectively. The 1st through 5th On-line World Conference on Soft Computing in Industrial Applications were held every year. State-of -the-art industrial applications were presented at those conferences. There were a large number of application papers of soft computing in the related IEEE Transactions and other related journals for the past ten years.

Soft Computing (SC) is an evolving collection of methodologies, which aims to exploit tolerance for imprecision, uncertainty, and partial truth to achieve robustness, tractability, and low total cost. Therefore, soft computing provides an attractive opportunity to represent the ambiguity in human thinking with real life uncertainty. Fuzzy Logic (FL), Neural Networks (NN), and Evolutionary Computation (EC) are the core methodologies of soft computing. However, FL, NN, and EC should not be viewed as competing with each other, but synergistic and complementary instead. Soft computing (SC) has been theoretically developed for the past decade, since Dr. L. A. Zadeh proposed the concept in the early 1990s. Later, chaos computing and immune networks were added to explain so-called complex systems, cognitive distributed artificial intelligence, and reactive distributed artificial intelligence. . Evolutionary computation has been developed and modified for application to optimization for large-scale and complex systems. Data mining technology, which has been developed since the late 80s using heterogeneous technologies, including soft computing methods based on pattern recognition techniques, has been recently used for interpreting and understanding important associations for large-scale and complex systems hidden in large database The term intelligence has been frequently used by Dr. Zadeh's human like information processing(cognitive artificial intelligence) and by Dr. Fogel's bioinformatic information processing(reactive artificial intelligence): i.e. intelligent behavior is able to result from an ability to predict the environment coupled with the selection of an algorithm that permits the translation of each prediction into a suitable response. Intelligent information processing is considered to be emergent, self-organizing (adaptive) and interactive among human beings, environment and artificial objects with an

advanced learning method using a combination of perception and motion. Soft computing plays an important role in intelligent information processing.

Soft computing is causing a paradigm shift (breakthrough) in engineering and science fields since soft computing can solve problems that have not been able to be solved by traditional analytic methods (tractability). In addition, SC yields rich knowledge representation (symbol and pattern), flexible knowledge acquisition (by machine learning from data and by interviewing experts), and flexible knowledge processing (inference by interfacing between symbolic and pattern knowledge), which enable intelligent systems to be constructed at low cost (high machine IQ and low cost). Tractability, high machine IQ and low cost enable industrial systems to be innovative.

Soft computing is considered to create new computational capabilities combining or fusing each soft computing methodology (hybrid system). This talk starts with the history of neuro-fuzzy control. Then, I will present the features of neuro-fuzzy hybrid systems. Lastly, I will introduce our developed novel fast and accurate fuzzy neural network for control and diagnosis of nonlinear dynamic systems using general parameter learning and adaptation.

Anticipatory Computing with Autopoietic and (M, R) Systems

Juan Carlos Letelier, Gonzalo Marín, Jorge Mpodozis and Jorge Soto-Andrade

Departamento de Biología

Facultad de Ciencias

Universidad de Chile

letelier@uchile.cl

Abstract. From the many attempts to produce a conceptual framework for the organization of living systems, the notions of (M,R) systems and Autopoiesis stand out for their rigor, their presupposition of the circularity of metabolism, and the new epistemologies that they imply. From their inceptions, these two notions have been essentially disconnected because each has defined its own language and tools. Here we demonstrate the existence of a deep conceptual link between (M,R) systems and Autopoietic systems. This relationship permits us to posit that Autopoietic systems, which have been advanced as capturing the central aspects of living systems, are a subset of (M,R) systems. This result, in conjunction with previous theorems proved by Rosen, can be used to outline a demonstration that the operation of Autopoietic systems cannot be simulated by Turing machines. This demonstration is rather delicate as it involves four different aspects: a) “the central theorem of Rosen” involving the stability of metabolic networks with circular organization, b) the use of the theory of Categories to model systems with circular organization, c) a preliminary proof that such systems can not be computable “a la Turing” and d) an identification between Autopoietic and (M,R) systems. This work will analyze with detail the mathematics behind Rosen central theorem as well as the implications for computing of the intrinsic anticipatory behavior of Autopoietic systems. The unfamiliar computing aspects of Autopoietic system arise from the internal reference frame of the controller. The control is done by the logic of the maintenance of circular organization in the presence of structural coupling. (see also paper in this book)

This page intentionally left blank

Section 2

Neural Systems

This page intentionally left blank

The use of a Back Propagation Neural Network to determine the load distribution on a component

R. Amali, J. Vinney, S. Noroozi, V. Patel

University of the West of England

Faculty of Computing, Engineering and Mathematical Sciences

Frenchay Campus, Coldharbour Lane, BS16 1QY, Bristol, U.K

Abstract. A method that combines a Back Propagation Neural Network (BPNN) with the data obtained using Finite Element Analysis (FEA) is introduced in this paper as an approach to solve inverse problems. This paper presents the feasibility of this approach. It demonstrates that the method approach works under laboratory or controlled conditions. FEA results are used to train the BPNN. The component used is a simple cantilever plate resembling an aircraft wing. Once trained, the approximate load distribution solution to any problem, bound by the training envelope, can be obtained quickly and accurately.

1. Introduction

The monitoring of in-service loads on many components has become a routine operation on simple and well-understood cases. However, as soon as the complexity of the structure increases, so does the difficulty in obtaining an acceptable understanding of the real loading. The collection of accurate and reliable load distribution data on aircraft components and structures is particularly difficult. However, this data is vital not only for design and development purposes, but also in order to determine overall structural integrity. It is generally accepted that for aircraft structures it is almost impossible to achieve acceptable levels of load prediction using the existing tools and techniques.

Finite Element Analysis and experimental stress analysis are two techniques widely used in industry to determine a structure's response to a known loading scenario. When applied to complex problems, these techniques can become expensive and time-consuming. In addition, when used in conjunction, they often generate conflicting results. The analysis of complex structures using the above processes usually involves a great deal of iteration and refinement in order to achieve acceptable results. Noroozi *et al* [1] showed that it is possible to reduce this solution time by interfacing these methodologies with more modern mathematical methods in order to create a hybrid analysis tool. A hybrid approach can be applied to both direct and inverse problems, both of which have been difficult to model, in the past.

2. Inverse Problem Method

The Inverse Problem Method (IPM) is based on accurately calculating the external loads or boundary conditions that can generate a known amount of stress or displacement at pre-determined locations on a structure. This is particularly useful when determining more realistic aerodynamic loads such as those encountered in flight. This detailed knowledge can lead to optimised lightweight structural designs.

Unlike parameters such as altitude or speed, the actual in-service loading on a wing is extremely difficult to quantify. However, it is easy to obtain information such as the surface strains, which can be used as the input to a network in order to determine the load distribution.

3. Artificial Neural Network

In general, Artificial Neural Networks (ANN) were developed from the study of the biological nervous system. In other words ANNs are an attempt to create machines which behave like the human brain by using components that behave like biological 'neurons'. These elements are all interconnected and work in conjunction with each other - hence the development of the name 'Neural Networks'.

A Neural Net involves large numbers of sensing and processing nodes that continuously interact with each other. Between each pair of nodes sits a non-linear transfer function such as a numerical weight, which helps to determine the final conclusion. As the neural network learns through the input of data, the weights are modified (or updated) for a better solution. Ultimately, the network learns the solution itself.

4. Modified BPNN

In order to change the weights in the right direction, a new technique with *two-momentum parameters* has been developed. The weights from three previous training patterns were saved and used to update weights using the formulae shown below:

$$W(k+1) = W(k) - \mu \partial Ex / \partial W \\ + 0.5\alpha_1(W(k) - W(k-1)) + 0.5\alpha_2(W(k-1) - W(k-2))$$

$$U(k+1) = U(k) - \eta \partial Ex / \partial U \\ + 0.5\beta_1(U(k) - U(k-1)) + 0.5\beta_2(U(k-1) - U(k-2))$$

Where

$$E_x = \frac{1}{2} \sum_{k=1}^K (d_k - y_k)^2$$

Where μ , η are positive valued scalar gain or learning rate constants and, $\alpha_1, \alpha_2, \beta_1$ and β_2 are momentum parameters, all less than 1. U and W are weights which connecting the input layers to hidden layer and hidden layer to the output layer, and d_k is desired output vector response and y_k is a vector of the actual system response.

5. Simulation problem

For preliminary investigations it was decided that a simple model of an aircraft wing would be used rather than an actual aircraft structural component. This facilitated quick and accurate FEA simulation to be carried out and relatively controlled experimental conditions for the experimental training sessions. Figure1 shows the idealised component made of plane sheet metal cut to shape and mounted as a tapered cantilever simulating the wing.

The location of strain gauges on the test piece was important in order to achieve acceptable results. It ensured that the Neural Network could easily interpret the data from

these known locations. It was important for the gauges to cover as wide an area of the component as possible in order to maximise the knowledge about the component behaviour. The pattern of gauge location is arbitrary and independent of the solution routine. Three channels of the data acquisition were reserved for the load cells and the other nine channels were used to collect strain values. To ensure the uniqueness of the solution it was decided that 10 strain gauges would be adequate.

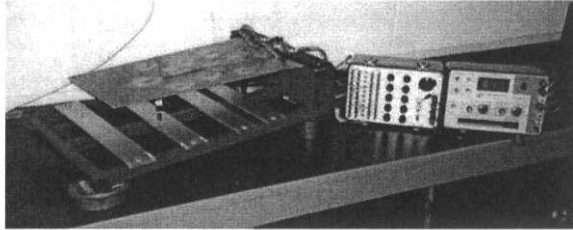


Figure1 – Idealised component

The location of the 10 strain gauges is listed in Table1.

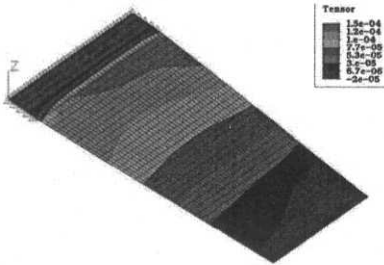


Figure2- FE model of the wing

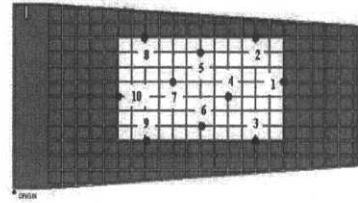


Figure3- Strain gauges on the wing

These positions are described in relation to the reference point located on the bottom of the component as shown in Figure 3. The flow chart in Figure 4 shows the training procedure using FEA data to train the network. An intelligent randomised problem generator is used to generate training data.

Table1- Position of the strain gauges

Gauge Number	X (mm)	Y (mm)
1	390.68	145.60
2	350.43	205.50
3	350.43	65.00
4	310.43	124.50
5	270.50	185.00
6	270.50	74.50
7	230.10	145.60
8	190.40	205.50
9	190.40	65.00
10	150.25	124.50

Once enough data has been gathered using FEA, covering the working envelope of the structure, training patterns can be introduced to the BPNN system. To check for convergence trial solutions are introduced to the system to measure the error. If the error is unacceptable, additional data will be generated and presented to the network. This is repeated until the error between trial solutions and those calculated by the network is acceptable. Once the error has been minimised the system has converged and the General Transfer Function (GTF) between input and output is created.

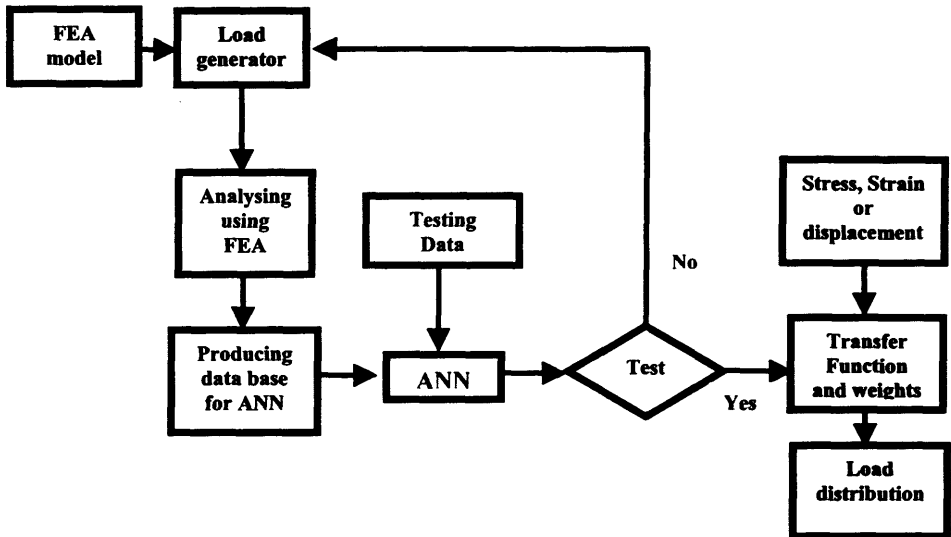


Figure 4 - Training procedure using FEA

6. Results and Discussion

This research has resulted in the creation of a general Inverse Problem Engine (IPE) for the structure under consideration. It is now possible to determine the position and magnitude of any loads from the surface strains. Graphs presented below in Figure5 show a selection of output from many test runs of the system. The results must be considered on a test basis. Test 1, 2, 3 and 4 compare the actual applied load with that obtained from the IPE; next to it is the comparison between the X and Y co-ordinates of the location for the applied load and those calculated using our real-time IPE. As shown, the error is very low and well within normally acceptable levels for all practical purposes.

Inherently, and due to the characteristics of any ANN, larger percentage errors are expected at low absolute output values and lower relative errors are expected at high absolute values. This is a desirable feature of the network approach, as the solution will always lie within a predefined error band. This gives a high probability of getting a result of acceptable accuracy but in real-time and independent of the size of the problem. This is an excellent feature of this IPE, which can be embedded as a central part of the system as it monitors stress or the overall condition of in-service components. This can be achieved in real-time providing an accurate load history for a component.

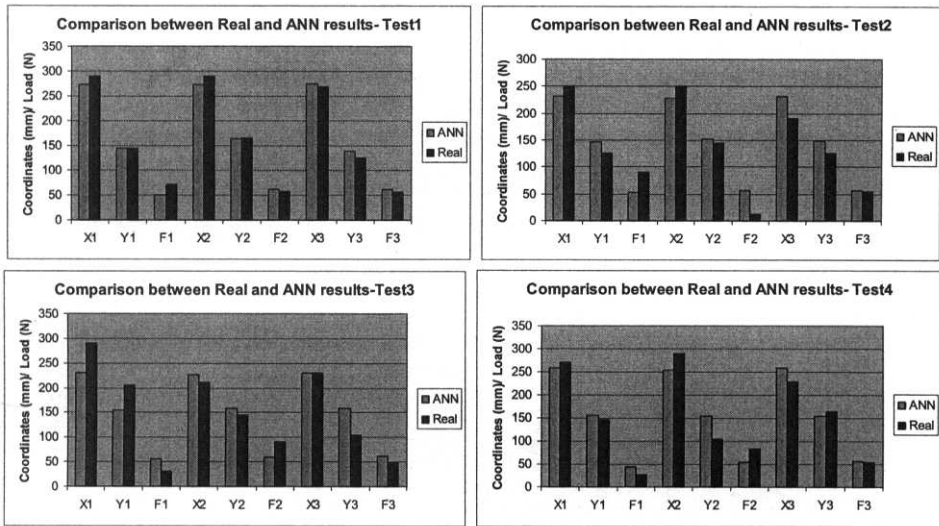


Figure 5- Comparison between actual and calculated load magnitudes and positions

7. Conclusions

In conclusion, it has been shown that this approach can be used to accurately predict loads on a simplified wing model. This powerful approach can offer high-speed solutions with the benefit of relatively high accuracy. The approach can provide FEA, or experimental quality, solutions in 'real-time' and can be part of a sophisticated embedded system. The system is cheap and robust, and once the transfer function for the real-time inverse problem engine has been created it can work as an independent algorithm.

In this study, a simplified model of an aircraft wing has been used to demonstrate the approach. Once trained for this component, the network can determine the approximate solution to any problem, bound by the training envelope, quickly and with a high degree of accuracy.

The ultimate goal of this research at UWE is to develop an IPE for the solution of any structural response problem. An Artificial Neural Network has been trained with a series of training data, and a GTF between the input and output has been generated. The GTF enables the magnitude and the position of loads acting on a complex geometry to be instantaneously quantified. Both the experimental and simulation results have been used to train the network.

8. References

1. R. Amali S. Noroozi, J. Vinney "The use of Back propagation Neural networks for the solution of Finite Elements problems, EANN 2000 17-19th July 2000.
2. Mitchell T.M. (1997), Machine Learning, Mc Graw-Hill International Edition.
3. Laurene F. Fundamental of Neural Networks, Prentice Hall International Editions
4. Romer M.J., Hong C. and Hesler S.H. Machine health monitoring and life management using finite element based and neural network, Journal of Engineering for Gas Turbines and Power. Oct. 1996, vol. 118.

5. Murase H. and Kayama S. The Finite Element Neural Network Application to Plant Factory, IFAC 12th terminal world conference, Sydney, Australia, 1993.
6. Low T.S., Chao B.I. The use of finite elements and neural networks for solution of inverse electromagnetic problems. IEEE Transcription of Magnetic, vol. 28, no 5, Sep. 92.
7. Hirishi O. Neural Network based parameters estimation for non-linear finite element analysis, Engineering Computation, vol. 15, 1998, pp. 103-138.
8. Segerland L.J. Applied Finite Element Analysis, John Wiley and Sons.
9. G.Xu, S.Noroozi, G.Littlefair, R.Penson, R.Callan, 1999. "A Finite Element Based Neural Network Model", MESM' 99, Society for Computer Simulation International European Council, University of Jordan. March 1999.
10. Xian Jiatong University, Xi'an, China. G.Xu, S.Noroozi, G.Littlefair, R.Penson, R.Callan 1999 "An HNN for Finite Element Analysis" ICAMT'99 , The International conference on Advanced Manufacturing Technology, June 1999.
11. S. Noroozi, J. Vinney. G. Xu, G. Littlefair, "A hybrid ANNT and FEA approach for potential problems". Conference on Advanced Engineering Design, Delft University, April 2000.
12. J. L. Wearing, A. Sheikh, S. Noroozi. A Boundary Element, Super Element Formulation for Potential Problems. 6th International Conference in BEM in Engineering Swansea, 11- 15 July 1989,
13. A. Sheikh, S. Noroozi. .Application of Combined Finite Element and Boundary Element Technique for Thermal Analysis. 11th International Conference on BEM in Engineering, Massachusetts, USA, 29 - 31 August 1989. J. L. Wearing,
14. Swansea. J. L. Wearing, A. Sheikh, S. Noroozi. Combined FE, BE Methods for Thermal Analysis. 6th International Conference on Numerical Method in Laminar and Turbulent Flow 11 - 15 July 1989,

Performance-guided Neural Network for Rapidly Self-Organising Active Network Management

Sin Wee LEE¹, Dominic PALMER-BROWN¹,
Jonathan TEPPER², Christopher ROADKNIGHT³

¹*Leeds Metropolitan University, Computational Intelligence Research Group,
Beckett Park, LS6 3QS Leeds, UK.*

²*The Nottingham Trent University, Department of Computing and Mathematics,
Burton Street, NG1 4BU Nottingham, UK.*

³*BTexact Technologies, BT Adastral Park, Martlesham Heath, IP5 3RE Ipswich, UK.*

Abstract. A neural network architecture is introduced for the real-time learning of input sequences using external performance feedback. The target problem domain suggests the use of Adaptive Resonance Theory (ART) networks [1] that are able to function in a robust and fast real-time adaptive active network environment, where user requests and new proxylets (services) are constantly being introduced over time [2,3]. The architecture learns, self-organises and self-stabilises in response to the user requests and maps the requests according to the types of proxylets available. However, the ART1 architecture and the original algorithm are modified to incorporate an external feedback mechanism whereby the performance of the system is fed into the network periodically. This modification, namely the 'snap-drift' algorithm, uses fast convergent, minimalist learning (snap) when the overall network performance has been poor and slow learning (drift towards user request input patterns) when the performance has been good. A key concern of the research is to devise a mechanism that effectively searches for alternative solutions to the ones that have already been tried, guided simultaneously by the input data (bottom-up information) and the performance feedback (top-down information). Preliminary simulations evaluate the two-tiered architecture using a simple operating environment consisting of simulated training and test data.

1. Introduction

Network users continually demand more services, causing the management of existing networks to become very expensive and computationally intensive. The deployment of new services is also restricted by slow standardisation, the difficulties of adding new functionality onto the existing internetworking equipment and the complexity of the existing network. An ideal network infrastructure would be one that is as simple as possible, removing duplication of management overhead, minimising signalling and moving towards an optimum (hands-off) network. 'Active networking' [2] aims to achieve this.

In this paper, we extend this area of research and propose the use of a Performance-guided Adaptive Resonance Theory (PART) as a means of finding and optimising a set of conditions that produce an optimum proxylets selection of the Execution Environment for Proxylets (EEP) which contain the most frequently requested proxylets (services).

1.1 Active Networking

The first proposal of active networking was introduced by Tennenhouse & Wetherall [2] to increase network flexibility and to overcome the complexity of current network systems by adding active service codes (unique software programs to perform a specific service) to the user-request packet header. Active networks provide extra mechanisms that enable clients to add software programs and policies of their own, rather than being tied to those provided by the network operators at present. The first approach, known as the *capsule* approach is used to enable the active service codes to run on the network devices, such as servers, that the packets encounter. Despite the difficulties of the approach [4], it has highlighted the important requirements for a feasible active network and encouraged other researchers to develop better alternatives to resolve them.

The first alternative, *active bridging* [5] was introduced in an attempt to resolve the issue of restricted code size, and the potential of this alternative gives network operators the freedom to choose appropriate active service codes of their own, which has been tested. However, one limitation associated with this approach is that the range of flags cannot be large, as the space available in the transport layer header of the Internet protocol is insufficient to accommodate the flags indicating the desirability of running a programme.

The second alternative, known as the *programmable networks* [6], is where clients of the network can download their own active service codes onto network devices before using their application. However this approach makes the assumption that the clients know which devices will be used whereas in reality, clients usually have no idea where his/her packets go because the flow of packets usually follow a number of different routes through the network.

The Application Layer Active Network (ALAN) is the third alternative and is assumed to be the most realisable. The ALAN architecture was first proposed by Fry and Ghosh [3] to enable the user to supply JAVA based active service code known as *proxylets* that runs on an edge system (Execution Environment for Proxylet – EEP) provided by the network operator. An edge system is a piece of hardware connected to the network, which is dedicated to managing proxylets within a specific network domain. The purpose of the architecture is to enhance the communication between servers and clients using the EEPs that are located at optimal points of the end-to-end path between the server and the clients without dealing with the current system architecture and equipments. This approach relies on the redirecting of selected request packets into the EEP, where the appropriate proxylets can be executed to modify the packets contents without impacting on the router's performance and thus does not need any additional standardisations.

1.2 Related Work

In the original ALAN proposal, the main objective was to propose an active service network architecture and not to put much emphasis on the management of the network. Initially, the management system supports a conventional management agent interface [7,8] that responds to high-level instructions from the system operators. However, since ALAN provides active services that are unbounded in both scale and functions and with an

enormous range of services being developed and evolved at an unprecedented rate, it is necessary to combine the active services with a highly automated and adaptive management and control solution. Recently, a novel adaptive approach to this problem of automated network management was introduced by Marshall and Roadknight from British Telecom (BT) Research Laboratories [9], which solves this problem with some success. The approach taken utilises a distributed Genetic Algorithm (GA) based on the unique methods that bacteria use to transfer and share genetic material. This is done by programming the management system to export the network active service codes (proxylets) to the neighbouring EEP in order to improve its performance, and also to remove or deactivate network active service codes that degrade performance. This means that only useful network proxylets will spread and poor or inactive network proxylets will cease to be executed until environmental (user requests) conditions change.

The algorithm was applied to the adaptive management solution and to the differentiated quality of service mechanism defined by Marshall & Roadknight [9,10] for ALAN and has shown promising results with respect to increasing the performance of the ALAN network.

In this paper, we describe an artificial neural network (ANN) based approach for the adaptive management of ALAN. The approach concentrates on harnessing the properties of ART networks, rather than applying the original ART network per se, in order to develop a novel ANN-based solution to adaptively select the appropriate proxylets available to the local EEPs, in response to continually changing input requests. A novel reinforcement-based learning algorithm is developed to improve the performance of the network under different commercial circumstances.

The remainder of this paper is structured as follows: Section 2 briefly describes the main feature of Adaptive Resonance Theory neural networks that are sufficient for the understanding of the rest of the paper. The architecture and the learning principles of the proposed architecture, called Performance-guided ART (PART) network, are described in Section 3. Experiments that we carried out on the novel reinforcement-learning algorithm on the Distributed P-ART module (see Figure 2) are discussed in Section 4, and conclusion and future work are discussed in Section 5.

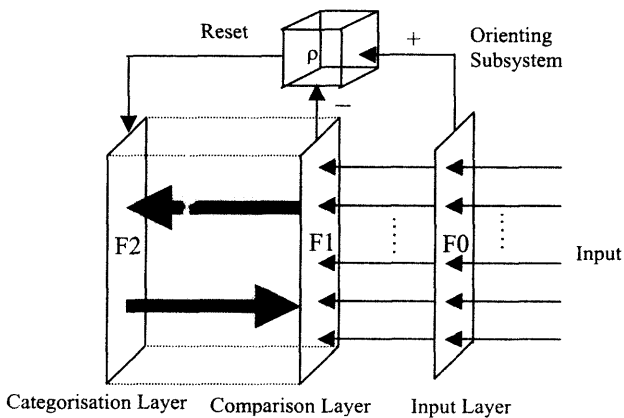


Figure 1: Architecture of ART1 network.

2. Adaptive Resonance Theory (ART) Neural Networks

ANNs are well known to be a useful alternative to the traditional Artificial Intelligence (AI) methods [11] such as tree search, graph search and back chaining for searching problems spaces. This is mainly due to their inherent parallelism, adaptability and ability to handle noisy and incomplete data. Although there have been a range of successful real-time applications with ANNs, the traditional neural network architectures and learning algorithms such as Multi-layered Perceptron (MLP) trained with the back-propagation algorithm [12,13] recurrent networks trained with back-propagation algorithm [14] and Kohonen Self-organising Map [15] are known to have several drawbacks when applied to real-time problem domains [16], such as long training times and the difficulty of understanding the effect of parameter changes on the environment as well as the knowledge encoded within the network. Furthermore, they typically have difficulty coping with complex, non-stationary environments and when they require offline training they are unsuitable for a constantly changing problem space.

The real-time and dynamic problems posed by the ALAN environment, suggests that the ANN solution required must be able to function in a robust, fast real-time adaptive network environment where user requests and new proxylets (services) are constantly being introduced over time. The focus of this research will therefore centre around a neural network solution that has automatically acquired knowledge of user input requests in real-time.

Adaptive Resonance Theory (ART) networks were first introduced to overcome the *stability – plasticity dilemma* [1,17,18] every learning system has to face. To illustrate the working of an ART network, consider an ART1 [1] network (see Figure 1) that is capable of fast and stable learning by clustering arbitrary binary input patterns using self-organization.

ART1 network is built up of 3 layers: the input layer (F0), the comparison layer (F1) and the categorisation layer (F2) with N, N, and M number of nodes, respectively. Each node of the input layer is connected to its corresponding node in the comparison layer via one to one, non-modifiable links. F1 and F2 layer is interconnected through weighted bottom-up and top-down connections that can be modified during the learning stage. The learning process of the network can be described as follows.

Upon the presentation of a binary input pattern I ($I_j \in \{1,0\}$, $j = 1,2,3, \dots, N$), the network attempts to categorize it by comparing it against the stored knowledge of the existing categories of each F2 node. This is achieved by calculating the bottom-up activation, which can be expressed as

$$T_i = \frac{|w_i \cap I|}{\beta + |w_i|} \quad (1)$$

Then the F2 node with the highest bottom-up activation, i.e. $T_i = \max \{T_i \mid i = 1,2, \dots, M\}$ is selected. If a category is found with the required matching level, known as vigilance level, where

$$\frac{|w_i \cap I|}{|I|} \geq \rho \quad (2)$$

where vigilance parameter $0 < \rho < 1$, F2 node J will enters into a resonance state and learns by modifying its prototype to keep only the critical features for the selected output category.

$$w_{ij}^{(new)} = \eta (w_{ij}^{(old)} \cap I) + (1 - \eta) w_{ij}^{(old)} \quad (3)$$

where η is the learning rate ($0 < \eta < 1$). All other weights in the network remain unchanged.

If no existing matching prototype is found, i.e. when the stored w_j do not match the input, then the winning F2 node is reset and another F2 node with the highest T_i is selected, whose prototype will be matched against the input, and so on. When no corresponding output category can be found, the network considers the input as novel, and generates a new output category that learns the current input. Essentially, it is a fast adaptive form of competition-based learning [16].

3. The Performance-guided ART (PART) Architecture

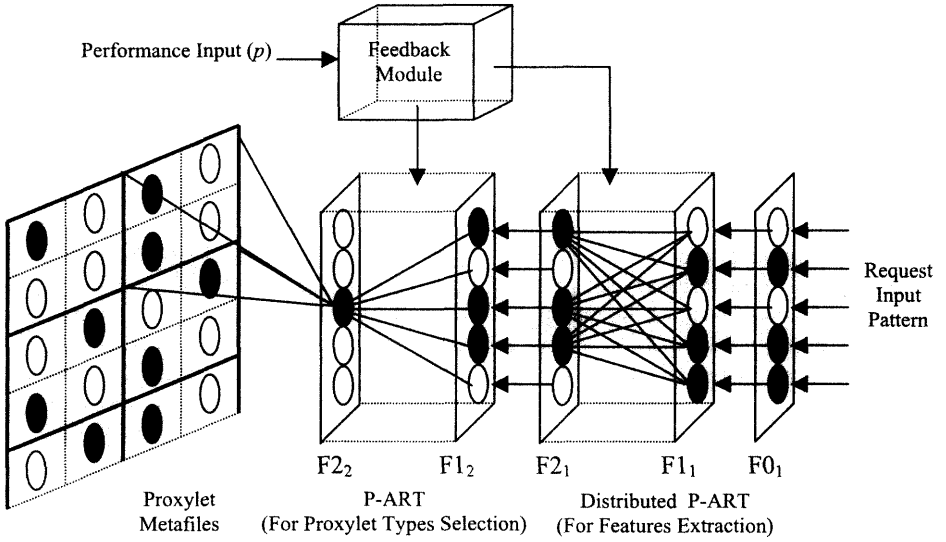


Figure 2: Architecture of the PART network.

The PART network we propose is a modular, multi-layered architecture that can be used to select available proxylets (services) in the networks, in response to continually changing input requests while trying to improve the overall performance of the network. It composed of 3 modules, a Distributed P-ART network, a P-ART network and a Kohonen Feature Map. The $F1_1 \leftrightarrow F2_1$ connections of the Distributed P-ART network and $F1_2 \leftrightarrow F2_2$ the P-ART are interconnected through weighted bottom-up and top-down connections that can be modified during the learning stage. For clarity, only the connections from the F1 layer to the active (winning) F2 node in each P-ART module are shown and the orienting subsystem is not included for simplicity. The $F0_1 \rightarrow F1_1$ and two P-ART modules connected through $F2_1 \rightarrow F1_2$ are unidirectional, one to one and non-modifiable. Each of the $F2_2$ nodes are hard-wired onto a specific pre-trained region of the Kohonen Feature map where similar available proxylets are spatially organised on the 2-D map according to their featural similarity (similar requests will appear in neighbouring map regions).

The working of the network can be summarised as follows: Upon the presentation of an input pattern at the input layer $F0_1$, the Distributed P-ART will learn to group the input patterns according to their general features using the novel learning principles, known as

'snap-drift' proposed in the next section. If no existing matching prototype is found, i.e. when the stored pattern templates is not a good match for the input, then the winning F_{2i} node is reset and another F_{2i} node is selected, whose pattern template will be matched against the input, and so on. When no corresponding output category can be found, the network considers the input as novel, and generates a new output category that learns the current input pattern.

The top three F_{2i} nodes are used as the input for the P-ART module where an appropriate proxylet type is selected accordingly. For the purpose of selecting the required proxylet, the proxylet type information indicated by the P-ART acts as a reference to pre-trained locations on the Kohonen-Feature Map, which represent specific proxylets. If the proxylet is unavailable, one of its neighbours is selected (the most similar alternative available).

In order to guide the network's learning, there is an external indicator of performance of the system as a whole. A general performance measure is used because there are no specific performance measures (or external feedback) in response to each *individual* network decision. This measure is used to enable the reselection of a proxylet types to occur in order to improve system performance.

3.1 Learning Principles

The essential departure of the PART learning principles from ART learning is the introduction of a mechanism that permit fast learning to occur repeatedly in order to find different solutions (sets of weights). In ART, once the fast learning has occurred, the weights are frozen. New patterns or circumstances will not result in new weights. In PART, however, slow clustering style learning is deployed in conjunction with fast minimalist learning, and the result is that the network can reorganise when new circumstances arise.

The learning process of the PART network is described as follows. Note that the novel learning principles described here are applied to the both P-ART modules in Figure 2.

3.1.1 Top-down Learning

The top-down learning of the network can be illustrated using the following equation:

$$w_{ij}^{(new)} = (1 - p) (I \cap w_{ij}^{(old)}) + p (w_{ij}^{(old)} + \beta (I - w_{ij}^{(old)})) \quad (4)$$

where

$w_{ij}^{(old)}$ = the top-down weights vectors at the start of the input presentation

p = performance parameter

I = binary input vectors

β = 'drift' constant

Initially, all the w_{ij} are set randomly to (0.99,1)

$$w_{ij}(0) = (0.99, 1) \quad (5)$$

Thus during learning, a simple distributed affect will be generated at the output layer of the network, with different patterns tending to give rise to different activations across F_2 . By substituting p in equation (4) with 0 for poor performance, equation (4) simplifies to:

$$w_{ij}^{(new)} = (I \cap w_{ij}^{(old)}) \quad (6)$$

Thus fast learning is invoked, causing the top-down weights to reach their new asymptote on each input presentation:

$$w_j \rightarrow I \cap w_j^{(old)} \quad (7)$$

In contrast, for excellent performance where $p = 1$, equation (4) can be simplified to:

$$w_{ij}^{(new)} = (w_{ij}^{(old)} + \beta (I - w_{ij}^{(old)})) \quad (8)$$

Thus, a simple form of clustering occurs at a speed determined by β .

It is assumed that there is a considerable interval between updates of p during which time new previously unseen requests are likely to appear. Equation (8), or indeed equation (4) whenever performance is not perfect, enables the top-down weights to drift towards the input patterns. New category node selection may occur for one of two reasons: as a result of the drift itself, or as a result of the drift enabling a further snap to occur (since drift has moved away from convergence) if performance p goes down. Essentially, the principle is that drift, by itself, will only results in slow (depending on β) reselection over time, thus keeping the network up-to-date without a radical set of reselections for exiting patterns. By contrast, snapping results in rapid reselection of a proportion of patterns to quickly respond to a significantly changed situation, in terms of the input vectors (requests) and/or of the environment which may require the same requests to be treated differently. For example, bandwidth and/or completion time may become critical at certain times. In this simulation, β is set to 0.5. This will provide the control for the drifting of the weights vector where eventually

With alternate episodes of $p = 0$ and $p = 1$, the characteristics of the learning of the network will be the joint effects of the equation (6) and (8). This joint effect can enable the network to learn using fast and convergent, snap learning when the performance is poor, yet be able to drift towards the input patterns when the performance is good.

3.1.2 Bottom-up Learning

In the simulations, the bottom-up weights w_{ji} are assigned initial values corresponding to the initial values of the top-down weights w_{ij} . This is accomplished by equation (9):

$$w_{ji}(0) = \frac{w_{ij}(0)}{1 + N} \quad (9)$$

where N = number of input nodes

By selecting this small initial value of w_{ji} , the network is more likely to select a previously learned category node that to some extent matches the input vector rather than an uncommitted node. For learning,

$$w_{ji}^{(new)} = \frac{I \cap w_{ji}^{(old)}}{|I \cap w_{ji}^{(old)}|} \quad (10)$$

This is fast, convergent learning.

4. Network Simulations

This section presents the results of a number of simulations performed on the distributed P-ART module to emulate possible conditions and thus evaluate the behaviour of the module in permitting fast learning to occur repeatedly in order to find different solutions. Three simulations were performed:

- (i) Long period without external performance feedback (performance $p = 0$)
- (ii) Long period where there is a constant supply of good performance feedback (performance $p = 1$)
- (iii) Long periods of poor and good feedback being available on alternate time steps (performance $p = [0, 1]$)

These simulations illustrate the influence of performances feedback toward the on-line learning of the network. Here the vigilance value is taken at $\rho = 0.5$. Whilst the vigilance value is not of critical important, it needs to be sufficiently high so that in combination with suitable initial weight values, as shown in equation (5), a range of winning nodes will emerged. Ultimately, however, the vigilance parameter is not considered the only mechanism for handling mismatch [19,20]. The test patterns consist of 100 input vectors. Each test pattern characterizes the features/properties of a realistic network request, such as bandwidth, time, file size, loss and completion guarantee. These test patterns were presented in random order for 10 epochs where the performance, p , is set to the different possible environmental conditions (long period of poor performance, long period of good performance and alternate performance feedback). This on-line continuous random presentation of test patterns simulates the possible real world scenario where the order of patterns presented is random so that a given network request might be repeatedly encountered while others are not used at all. Furthermore, no input pattern is removed from the list of possible requests after it is encountered. The network operates according to the principle that all network requests can be classified. This is achieved by initialising a sufficient number of output nodes so that at least one is always a good match. During simulation, the network will select 3 output nodes during learning (the 3 node with the highest activations) to provide a distributed output category of the network request type that is then fed into the next stage of the hierarchical system for the proxylet type selection.

4.1 Simulation Results

The results of the simulations is summarised in Table 1 and Table 2. These tables show some selected inputs with the prototype they learned to show the general feature extraction (clustering) of the test patterns. Furthermore, these tables also show the influences of the performance feedback and the effects of the 'snap-drift' algorithm towards the learning of the prototype.

In simulation (i), the networks learned to classify the test patterns rapidly and accurately. Almost all of the input vectors stabilised within 4 epochs of the training simulations. During learning, the network categorised the 100 request patterns into 47 categories. Note that each row of the input pattern represents the binary representation of a user network request, as mentioned. The general features of the input patterns are distributed into 3 winning activation nodes. In Table 1, each asterisk * represents non-zero value of a weight. The 3 winning activation nodes will then be used as inputs into the P-ART for proxylet type selection (see Figure 2). The redundancy across the three prototypes (top-down templates) in Table 1 shows the features-cells that play a role in determining those winning activation nodes for the proxylets type selection stage.

In simulation (ii), the network learned to classify the test input vectors more gradually than in Simulation (i). This is because during learning, the weights are drift towards the input vectors, in comparison with the snap effect in case 1. Furthermore, each of the learned node is not completely committed, causes less category to be formed. In this simulation, the network have categorised the 100 test input vectors into 25 categories.

Table 1: General feature extraction of the input patterns.

Input 2	Input 24	Input 66	Input 100
** **** **** ** **	**** **** ** **** *****	* ** ***** * *****	***** * * ***** *
Top-down Templates			
* *** ** * *	*** *** * ***	* ** * ***	** * ****
* *** ** * *	** *** * ** *	* * * ***	*** * **** *
* *** *** *	** *** * * *	* ** **	*** * *** *

Simulation (iii) provides a dynamics view of the network where it is continuously learning throughout the process with p alternating between good and bad (see Table 2). The overall picture therefore is an ongoing partial reselection. At the beginning of the simulation, the performance p is first initialised to 0 to invoke system learning. At the first epoch, the network initiates fast learning until the first performance feedback is fed into the network. Here, it is assumed that the performance of the network is measured after every simulation epoch. Since the network is fed with the dummy performance values of (0,1) alternating, the weights started to drift toward the input vector at the start of the second epoch ($p = 1$). Some reselections of old/new patterns happen as a result of this second epoch.

The re-selection of the output node in attempt to improve the network performance started when the network go from good performance to bad performance phase in the fifth epoch when the input pattern is encountered again. In the fifth epoch ($p = 0$), some of the inputs vectors will select different output nodes, either previously committed or uncommitted nodes, depending on the drifting of the top-down weights they learned on the previous epoch. Furthermore, since more then one output node is selected for learning at the beginning of the simulation, partial reselection is possible where two output nodes out of the 3 output nodes previously selected is re-learned.

During this simulation, the 100 test input vectors are categorised into 50 categories.

Table 2: Influence of the ‘snap-drift’ algorithm towards the learning of the network in Simulation (iii)

<div> <div>Input 70</div> <div> <div>***</div> <div>*****</div> <div>*</div> <div>*****</div> <div>****</div> </div> </div>			
Top-down Templates			
	Node 38	Node 79	Node 87
Epoch 1 (Performance $p = 0$)	<div> <div>*</div> <div>**</div> <div>*</div> <div>**</div> <div>**</div> </div>	<div> <div>**</div> <div>*</div> <div>***</div> <div>***</div> </div>	<div> <div>***</div> <div>**</div> <div>***</div> <div>***</div> </div>
	Node 85	Node 116	Node 134
Epoch 2 (Performance $p = 1$)	<div> <div>**</div> <div>****</div> <div>*</div> <div>**</div> <div>*</div> </div>	<div> <div>****</div> <div>*</div> <div>****</div> <div>*</div> </div>	<div> <div>***</div> <div>**</div> <div>*</div> <div>**</div> <div>**</div> </div>
	Node 95	Node 103	Node 127
Epoch 5 (Performance $p = 0$)	<div> <div>**</div> <div>***</div> <div>**</div> <div>**</div> </div>	<div> <div>**</div> <div>**</div> <div>*</div> <div>***</div> </div>	<div> <div>**</div> <div>***</div> <div>**</div> <div>*</div> </div>
	Node 81	Node 103	Node 145
Epoch 9 (Performance $p = 0$)	<div> <div>**</div> <div>*</div> <div>***</div> </div>	<div> <div>**</div> <div>*</div> <div>***</div> </div>	<div> <div>**</div> <div>*</div> <div>***</div> </div>

5. Conclusions and Future Work

In this paper, we have introduced a network architecture containing modules that combine ART style learning with clustering according to performance feedback. It is capable of stable learning of the network input request patterns in real-time and is able to map them onto the appropriate proxylets available to the system. We have shown the plausibility of the ‘snap-drift’ algorithm, which is able to provide continuous real-time learning in order to improve the network performance, based on the external performance feedback. These system properties have been confirmed by the results obtained from the experiments performed using the Distributed P-ART module, which was evaluated using performance feedback scenarios.

We believe that the idea and the preliminary results presented here provide a platform for further research. The full architecture will be thus evaluated to explore its full potential in providing adaptive network management. There are also several ways to further explore the architecture and algorithm:

- Explore different ways of using the performance feedback, e.g. look at the network behaviour in changing of performance under different known commercial circumstances.
- Investigate bottom-up learning with performance feedback to improve the efficiency of the reselection process.
- More research will be carried out to explore the possibility of modifying or substituting the original ART mismatch process with a process that immediately finds the best match [19,20].

We hope that further experiment with the proposed PART network will enable us to further understand the network behaviour, and apply this network architecture and/or its learning principles to a range of different problems.

References:

- [1] G.A. Carpenter, S. Grossberg, A Massively Parallel Architecture for a Self-Organising Neural Pattern Recognition Machine, *Computer Vision, Graphics and Image Processing* **37** (1987a) 54 – 115.
- [2] D.Tennehouse, D. Wetherall, Towards an active network architecture, *Computer Communication Reviews*, **26(2)** (1996) 5 – 18.
- [3] M. Fry, A. Ghosh, Application Layer Active Network, *Computer Networks*, **31(7)** (1999) 655-667.
- [4] I.W. Marshall, Active Networks – Making the next-Generation Internet Flexible, *British Telecommunication Engineering*, **18** (1999) 2 – 8.
- [5] D.S. Alexander et al, Active Bridging, *Computer Communication Review*, **26(2)** (1996) 101-111.
- [6] A.T. Campbell et al, Open Signalling for ATM, Internet and Mobile Networks, *Computer Communication Review*, **29(1)** (1999).
- [7] I.W. Marshall et al, Application Layer Programmable Internetwork Environment, *British Telecom Technology Journal*, **17(2)** (1999) 82 – 94.
- [8] I.W. Marshall et al, Active Management of Multiservice Networks, *Proceedings of IEEE NOMS2000*, pp. 981-983.
- [9] I.W. Marshall, C.M. Roadknight, Differentiated quality of service in application layer active networks. In: H. Yasuda (ed.), *Active Networks, Second International Working Conference, LNCS 1942* Springer-Verlag, Japan, 2000, pp. 358 – 370.
- [10] I.W. Marshall, C.M. Roadknight, Adaptive Management of an Active Services Network, *British Telecom Technology Journal*, **18(4)** (2000) 78 – 84.
- [11] L.V. Fausett, *Fundamentals of Neural Networks*, Prentice-Hall, New Jersey, 1994.
- [12] D.E. Rumelhart, G.E. Hinton, R.J. William, Learning Internal Representations by Error Propagations. In: D.E. Rumelhart, J.L. McClelland (eds.) *Parallel Distributed Processing*, **1(8)** (1986a). Reprinted in Anderson & Rosenfeld, 1988, pp. 675-695.
- [13] D.E. Rumelhart, G.E. Hinton, R.J. William, Learning Representations by Back-Propagating error, *Nature*, **323** (1986b) 533 – 536. Reprinted in Anderson & Rosenfeld, 1988, pp. 696 – 699.
- [14] J.F. Kolen, S.C. Kremer, *A Field Guide to Dynamical Recurrent Networks*, IEEE Press, New York, 2000.
- [15] T. Kohonen, Self-organized Formation of Topologically Correct Feature Maps, *Biological Cybernetics*, **43** (1982) 53 – 69. Reprinted in Anderson and Rosenfeld, 1988, pp. 511-521.
- [16] G.A. Carpenter, S. Grossberg, The ART of Adaptive Pattern Recognition by a Self-Organising Neural Networks, *IEEE Computer*, **21(3)** (1988) 77 – 88
- [17] S. Grossberg, Adaptive Pattern Classification and Universal Recoding. I. Parallel Development and Coding of Neural Feature Detectors, *Biological Cybernetic* **23** (1976a) 121 – 134
- [18] S. Grossberg, Adaptive Pattern Classification and Universal Recoding. II. Feedback, Expectation, Olfaction, and Illusions, *Biological Cybernetic* **23** (1976b) 187 - 202
- [19] D. Palmer- Brown, High Speed Learning in a Supervised, Self Growing Net, In: I. Aleksander and J. Taylor (eds.), *Proceeding of ICANN 92*, **2** (1992) 1159 – 1162, ISBN 0 444 894888, 1992, Brighton.
- [20] S. Barker, H. Powell, D. Palmer-Brown, Size Invariant Attention Focusing (with ANNs), *Proceeding of International Symposium on Multi-Technology Information Processing*. 1996.

An Automated Hybrid Reasoning System for Forecasting

Florentino Fdez-Riverola and Juan M. Corchado

*Dpto. de Informática, University of Vigo,
E.S.E.I., Campus Universitario As Lagoas s/n., 32004, Ourense, Spain
riverola@uvigo.es*

*Dpto. de Informática y Automática, University of Salamanca,
Facultad de Ciencias, Plaza de la Merced, s/n., 37008, Salamanca, Spain
corchado@usal.es*

Abstract. A hybrid neuro-symbolic problem solving model is presented in which the aim is to forecast parameters of a complex and dynamic environment in an unsupervised way. In situations in which the rules that determine a system are unknown, the prediction of the parameter values that determine the characteristic behaviour of the system can be a problematic task. The proposed system employs a case-based reasoning model to wrap a growing cell structures network, a radial basis function network and a set of Sugeno fuzzy models to provide an accurate prediction. Each of these techniques is used in a different stage of the reasoning cycle of the case-based reasoning system to retrieve, to adapt and to review the proposed solution to the problem. This system has been used to predict the red tides that appear in the coastal waters of the north west of the Iberian Peninsula. The results obtained from those experiments are presented.

1 Introduction

Forecasting the behaviour of a dynamic system is, in general, a difficult task, especially when dealing with complex, stochastic domains for which there is a lack of knowledge. In such a situation one strategy is to create an adaptive system which possesses the flexibility to behave in different ways depending on the state of the environment. An artificial intelligence approach to the problem of forecasting in such domains offers potential advantages over alternative approaches, because it is able to deal with uncertain, incomplete and even inconsistent data. This paper presents a hybrid artificial intelligence (AI) model for forecasting the evolution of complex and dynamic environments that can be numerically represented. The effectiveness of this model is demonstrated in an oceanographic problem in which neither artificial neural network nor statistical models have been sufficiently successful.

Traditionally, CBR systems have been combined with other technologies like artificial neural networks, rule-based systems, constraint satisfaction problems and others, producing successful results to solve specific problems [1, 2]. Although, in general each specific problem and domain requires a particular solution, this paper proposes a CBR based solution for forecasting the evolution of a complex problem, with a high degree of dynamism for which there is a lack of knowledge, and for which an adaptive learning system is required. This

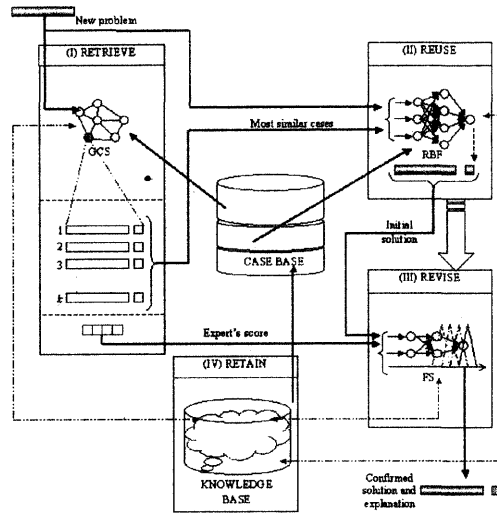


Figure 1: Hybrid neuro-symbolic system.

paper also presents, a method for automating the CBR reasoning process for the solution of problems in which the cases are characterized predominantly by numerical information.

Successful results have been already obtained with hybrid case-based reasoning systems [3, 4] and used to predict the evolution of the temperature of the water ahead of an ongoing vessel, in real time. The hybrid system proposed in this paper presents a new synthesis that brings several soft computing methods together (CBR, ANN and Fuzzy inferencing). The retrieval, reuse, revision and learning stages of the CBR system presented in this paper use the previously mentioned technologies to facilitate the CBR adaptation to a wide range of complex problem domains (for instance, the afore-mentioned red tides problem) and to completely automate the reasoning process of the proposed forecasting mechanism.

The structure of the paper is as follows: first the hybrid neuro-symbolic model is explained in detail; a case of study is then briefly outlined; the results obtained to date with the proposed forecasting system are analyzed, and finally, the conclusions and future work are presented.

2 The Hybrid CBR based Forecasting System

In this context of the previously presented domains, in order to forecast the value of any variable, a problem descriptor should be generated. A problem descriptor is composed of a vector with the variables that describe the problem and the solution. In this case, this vector holds numerical variables.

Figure 1 illustrates the relationships between the processes and components of the hybrid CBR system. In general, we can say that the forecast values are obtained using a neural network enhanced hybrid case-base reasoning system. The diagram shows the technology used at each stage, where the four basic phases of the CBR cycle are shown as rectangles.

The retrieval stage is carried out using a Growing Cell Structures (GCS) ANN [5]. The GCS facilitates the indexation of cases and the selection of those that are most similar to

the problem descriptor. The reuse and adaptation of cases is carried out with a Radial Basis Function (RBF) ANN [6], which generates an initial solution creating a forecasting model with the retrieved cases. The revision is carried out using a group of pondered fuzzy systems that identify potential incorrect solutions. Finally, the learning stage is carried out when the real value of the variable to predict is measured and the error value is calculated, updating the knowledge structure of the whole system. Now we present the working cycle of the CBR system illustrated in Figure 1.

When a new problem is presented to the system a new problem descriptor (case) is created, and the GCS neural network is used to recover from the case-base the k most similar cases to the given problem (identifying the class to which the problem belongs).

In the reuse phase, the values of the weights and centers of the RBF neural network used in the previous forecast are retrieved from the knowledge base. These network parameters together with the k retrieved cases are then used to retrain the RBF network and to obtain an initial forecast. During this process the values of the parameters that characterize the network are updated.

CBR-STAGE	Technology	Input	Output	Process
Retrieval	GCS network	Problem descriptor	k cases Expert's score	All the cases that belong to the same class to which the GCS associates the problem case are retrieved.
Reuse	RBF network	Problem descriptor k similar cases	Initial solution	The RBF network is retrained with the k retrieved cases.
Revision	Fuzzy systems	Problem descriptor Expert's score Initial solution	Confirmed solution	Different fuzzy systems are created using the RBF network configuration with different degrees of generalization.
Retrain	GCS network RBF network Fuzzy systems	Problem descriptor Forecasting error	Configuration parameters of the GCS network, RBF network and Fuzzy systems	The configurations of the GCS network, the RBF network and the Fuzzy subsystems are updated according to the accuracy of the forecast.

Figure 2: Summary of technologies employed by the hybrid model.

In the revision phase, the initial solution proposed by the RBF neural network is modified according to the response of the fuzzy revision subsystem (a set of fuzzy models). Each fuzzy system has been created from the RBF network using neuro-fuzzy techniques [7] as it will be seen later on.

The revised forecast is then retained temporarily in the forecast database. When the real value of the variable to predict is measured, the forecast value for the variable can then be evaluated, through comparison of the actual and forecast value and the error obtained. A new case, corresponding to this forecasting operation, is then stored in the case-base. The forecasting error value is also used to update several parameters associated with the GCS network, the RBF network and the fuzzy systems.

2.1 Growing Cell Structures Operation

When dealing with complex, dynamic, stochastic problems that can be numerically represented, the decision of what retrieval model is better to use for a CBR that need to be completely automated is not a trivial task. The main characteristics that should have the retrieval model are: an adequate the adaptation capacity to the problem domain and strong learning

capability. In such a situation, GCS neural networks offer several advantages over other approaches and well known metrics.

To illustrate the working model of the GCS network inside the whole system, a two-dimensional space will be used, where the cells (neurons) are connected and organized into triangles [5]. Each cell in the network (representing a generic case), is associated with a weight vector, w , of the same dimension that the problem descriptors stored in the case-base. At the beginning of the learning process, the weight vector of each cell is initialized with random values. The basic learning process in a GCS network consists of topology modification and weight vector adaptations carried out in three steps. The training vectors of the GCS network are the cases stored in the CBR case-base, as indicated in Figure 1.

In the first step of each learning cycle, the cell c , with the smallest distance between its weight vector, w_c , and the actual case, v_x , is chosen as the *winner cell* or best-match cell. The second step consists in the adaptation of the weight vector of the winning cells and their neighbours. In the third step, a *signal counter* is assigned to each cell, which reflects how often a cell has been chosen as winner. Growing cell structures also modify the overall network structure by inserting new cells into those regions that represent large portions of the input data, or removing cells that do not contribute to the input data representation.

Repeating this process several times, for all the cases of the case-base, a network of cells will be created. Each cell will have associated cases that have a similar structure and each cell will represent a class. These cells can be seen as a “prototype” that identifies a set of similar problem descriptors.

For each class identified by the GCS neural network a vector of values is maintained (see Figure 1). This vector (to which we will refer as “importance” vector) is initialized with a same value for all its components whose sum is one, and represents the accuracy of each fuzzy system (used during the revision stage) with respect to that class. During revision, the importance vector associated to the class to which the problem case belongs, is used to ponder the outputs of each fuzzy system. Each value of the importance vector is associated with one of the fuzzy systems. For each forecasting cycle, the value of the importance vector associated with the most accurate fuzzy system is increased and the other values are proportionally decreased. This is done in order to give more relevance to the most accurate fuzzy system of the revision subsystem.

Figure 3 provides a more concise description of the GCS-based case retrieval regime described above, where v_x is the value feature vector describing a new problem, confGCS represents the set of cells describing the GCS topology after the training, K is the retrieved set of most relevant cases given a problem and P represents the “importance” vector for the identified prototype.

The neural network topology of a GCS network is incrementally constructed on the basis of the cases presented to the network. Effectively, such a topology represents the result of the basic clustering procedure and it has the added advantage that inter-cluster distances can be precisely quantified. Since such networks contain explicit distance information, they can be used effectively in CBR to represent: (i) an *indexing structure* which indexes sets of cases in the case-base and, (ii) a *similarity measure* between case sets [8].

Another specially interesting fact is that the GCS network are structurally similar to the RBF network. The GCS network provides consistent classifications that can be used by the RBF network to auto-tuning its forecasting model.

In the light of all these reasons, the GCS neural network has been selected to solve the problem of the classification and indexing in our hybrid CBR based forecasting system.

```

procedure RETRIEVE (input:  $v$ , confGCS; output:  $K$ ,  $P$ )
{
00  begin.
01   $CD \leftarrow \emptyset$  /* vector of pairs (cell, distance) */
02  for each cell  $c \in \text{confGCS}$  do
03      compute_distance:  $dc \leftarrow \text{DIS}(v, w_c)$ 
04      assign_cell-distance-pair:  $CD \leftarrow (c, d_c)$ 
05  order_by_distance( $CD$ ) /* ascending */
06  for each pair  $p \leftarrow CD$  do
07       $K \leftarrow \text{get\_cases\_from\_cell}(p)$ 
08      if  $|K| > 0$  then
09          go_to_line 10 /* non-empty cell */
10  end.
}

```

Figure 3: GCS-based case retrieval.

2.2 Radial Basis Function Operation

Case adaptation is one of the most problematic aspects of the CBR cycle, mainly if we have to deal with problems with a high degree of dynamism and for which there is a lack of knowledge. In such a situation, RBF networks have demonstrated their utility as universal approximators for closely modelling these continuous processes [1].

Again to illustrate how the RBF networks work, a simple architecture will be presented. Initially, three vectors are randomly chosen from the training data set and used as centers in the middle layer of the RBF network. All the centers are associated with a Gaussian function, the width of which, for all the functions, is set to the value of the distance to the nearest center multiplied by 0.5 (see [6] for more information about RBF network).

Training of the network is carried out by presenting pairs of corresponding input and desired output vectors. After an input vector has activated each Gaussian unit, the activations are propagated forward through the weighted connections to the output units, which sum all incoming signals. The comparison of actual and desired output values enables the mean square error (the quantity to be minimized) to be calculated.

The closest center to each particular input vector is moved toward the input vector by a percentage a of the present distance between them. By using this technique the centers are positioned close to the highest densities of the input vector data set. The aim of this adaptation is to force the centers to be as close as possible to as many vectors from the input space as possible. The value of a is linearly decreased by the number of iterations until its value becomes zero; then the network is trained for a number of iterations ($1/4$ of the total of established iterations for the period of training) in order to obtain the best possible weights for the final value of the centers. A new center is inserted into the network when the average error in the training data set does not fall during a given period. There are different methods to identify the place where the new center will be inserted [6].

Figure 4 provides a more concise description of the RBF-based case adaptation regime, where v_x is the value feature vector describing a new problem, K is the retrieved set of most relevant cases, confRBF represents the previously configuration of the RBF network and f_i represents the initial forecast generated by the RBF.

The working model commented above together with their good capability of generalization, fast convergence, smaller extrapolation errors and higher reliability over difficult data, make this type of neural networks a good choice that fulfils the necessities of dealing with

```

procedure REUSE (input: v., K, confRBF; output: f)
{
00  begin.
01    while TRUE do /* infinite loop */
02      for each case c  $\in$  K do /* network adaptation using K cases */
03        retrain_network: error  $\leftarrow$  annRBF(c)
04        move_centers: annRBF.moveCenters(c)
05        modify_weights: annRBF.learn(c) /* delta rule */
06        if (error / |K|) < error_threshold then
07          go_to_line 8 /* end of infinite loop and adaptation */
08        generate_initial_forecast: f  $\leftarrow$  annRBF(v.)
09  end.
}

```

Figure 4: RBF-based case adaptation.

this type of problems. It is very important to train this network with a consistent number of cases. Such consistence in the training data set is guaranteed by the GCS network.

RBF networks can also be used to generate Fuzzy inference systems [9]. This characteristic has been used in this model for the automatic generation of the revision subsystem as it will be explained in the following section.

2.3 Fuzzy System Operation

The two main objectives of the proposed revision stage are: to validate the initial prediction generated by the RBF and, to provide a set of simplified rules that explain the system working mode. The construction of the revision subsystem is carried out in two main steps.

(i) First, a Sugeno-Takagi fuzzy model [10] is generated using the trained RBF network configuration (centers and weights) in order to transform a RBF neural network to a well interpretable fuzzy rule system [7]. Rule extraction from artificial neural networks is considered to be important due to the following reasons:

- Rule extraction provides artificial neural networks with an explanation capability, which makes it possible for the user to check on the internal logic of the system.
- Rule extraction helps to discover previously unknown dependencies in data sets and thus new knowledge about the system can be acquired.
- It is believed that a rule system with good interpretability improves the generalization ability of neural networks where training data are insufficient.

(ii) A measure of similarity is applied to the fuzzy system with the purpose of reducing the number of fuzzy sets describing each variable in the model. Similar fuzzy sets for one parameter are merged to create a common fuzzy set to replace them in the rule base. If the redundancy in the model is high, merging similar fuzzy sets for each variable might result in equal rules that also can be merged, thereby reducing the number of rules as well. When similar fuzzy sets are replaced by a common fuzzy set representative of the originals, the system's capacity for generalization increases.

In this model, the fuzzy systems are associated with each class identified by the GCS network, mapping each one with its corresponding value of the importance vector. There is

one importance vector for each class or prototype. These fuzzy systems are used to validate and refine the proposed forecast. Given a problem descriptor and a proposed forecast for it, each of the fuzzy inference systems that compose the revision subsystem generates a solution that is pondered according to the importance vector associated to the GCS class to which the problem belongs.

The value generated by the revision subsystem is compared with the prediction carried out by the RBF and its difference (in percentage) is calculated. If the initial forecast does not differ by more than 10% of the solution generated by the revision subsystem, the prediction of the RBF is supported and its value is considered as the final forecast. If, on the contrary, the difference is greater than 10% but lower than 30%, the average value between the value obtained by the RBF and that obtained by the revision subsystem is calculated, and this revised value adopted as the final output of the system. Finally, if the difference is greater or equal to 30% the system is not able to generate an appropriate forecast. These two thresholds have been identified after carrying out several experiments and following the advice of human experts.

Fuzzy systems provide a solution to the revision stage when dealing with complex problems, with a high degree of dynamism and for which there is a lack of knowledge. The exposed revision subsystem improves the generalization ability of the RBF network. The simplified rule bases allow us to obtain a more general knowledge of the system and gain a deeper insight into the logical structure of the system to be approximated.

The proposed revision method then helps us to ensure a more accurate result, to gain confidence in the system prediction and to learn about the problem and its solution. The fuzzy inference systems also provide useful information that is used during the retain stage.

2.4 Retain

As mentioned before, when the real value of the variable to predict is known, a new case containing the problem descriptor and the solution is stored in the case-base. The importance vector associated with the retrieved class is updated in the following way: The error percentage with respect to the real value is calculated. The fuzzy system that has produced the most accurate prediction is identified and the error percentage value previously calculated is added to the degree of importance associated with this fuzzy subsystem. As the sum of the importance values associated to a class (or prototype) has to be one, the values are normalized and the sum dividing up accordingly between them. When the new case is added to the case-base, its class is identified. The class is updated and the new case is incorporated into the network for future use.

3 A Case of Study: The Red Tides Problem

Red tides are the name for the discolourations caused by dense concentrations of the microscopic plants of the sea, the so-called phytoplankton. The discolouration varies with the species of phytoplankton, its pigments, size and concentration, the time of day, the angle of the sun and other factors. Red tides usually occur along the north west of the Iberian Peninsula in late summer and autumn [11]. The rapid increase in dinoflagellate numbers, sometimes to millions of cells per liter of water, is what is known as a *bloom* of phytoplankton (if the concentration ascends above the 100.000 cells per liter). The type of dinoflagellate in which this

Table 1: Variables that define a case.

Variable	Unit	Week
Date	dd-mm-yyyy	W_{n-1}, W_n
Temperature	Cent. degrees	W_{n-1}, W_n
Oxygen	milliliters/liter	W_{n-1}, W_n
PH	acid/basic	W_{n-1}, W_n
Transmittance	%	W_{n-1}, W_n
Fluorescence	%	W_{n-1}, W_n
Cloud index	%	W_{n-1}, W_n
Recount of diatoms	cel/liter	W_{n-1}, W_n
Pseudo-nitzschia spp	cel/liter	W_{n-1}, W_n
<i>Pseudo-nitzschia spp (future)</i>	<i>cel/liter</i>	W_{n+1}

study is centered is the pseudo-nitzschia spp diatom, causing of amnesic shellfish poisoning.

3.1 Forecasting Red Tides

In the current work, the aim is to develop a system for forecasting one week in advance the concentrations (in cells per liter) of the pseudo-nitzschia spp, the diatom that produces the most harmful red tides, at different geographical points. The approach builds on the methods and expertise previously developed in earlier research.

The problem of forecasting, which is currently being addressed, may be simply stated as follows:

- **Given:** a sequence of data values (representative of the current and immediately previous state) relating to some physical and biological parameters,
- **Predict:** the value of a parameter at some future point(s) or time(s).

The raw data (sea temperature, salinity, PH, oxygen and other physical characteristics of the water mass) which is measured weekly by the monitoring network for toxic proliferations in the CCCMM (Centro de Control da Calidade do Medio Marino, *Oceanographic Environment Quality Control Centre*, Vigo, Spain), consists of a vector of discrete sampled values (at 5 meters' depth) of each oceanographic parameter used in the experiment, in the form of a time series. These data values are complemented by data derived from satellite images stored on a database. The satellite image data values are used to generate cloud and superficial temperature indexes which are then stored with the problem descriptor and subsequently updated during the CBR operation. Table 1 shows the variables that characterize the problem. Data from the previous 2 weeks (W_{n-1}, W_n) is used to forecast the concentration of pseudo-nitzschia spp one week ahead (W_{n+1}).

A case-base was built with the above mentioned data. For this experiment, four fuzzy inference systems have been created from the RBF network, and they were initialized with a value of (0.25, 0.25, 0.25, 0.25) for each class (or prototype) in the GCS network. The RBF network used in the framework of this experiment, uses 18 input neurons, between three and fifty neurons in the hidden layer and a single neuron in the output layer, being the output of the network the concentration of pseudo-nitzschia spp for a given water mass.

The following section discusses the results obtained with the prototype developed for this experiment.

Table 2: Summary of results forecasting pseudo-nitzschia spp.

Method	OK	OK (%)	N. detect.	Fal. alarms	Aver. error (cel/liter)
CBR-ANN-FS	191/200	95,5%	8	1	26.043,66
RBF	185/200	92,5%	8	7	45.654,20
ARIMA	174/200	87%	10	16	71.918,15
Quadratic Trend	184/200	92%	16	0	70.354,35
Moving Average	181/200	90,5%	10	9	51.969,43
Simp. Exp. Smooth.	183/200	91,5%	8	9	41.943,26
Lin. Exp. Smooth.	177/200	88,5%	8	15	49.038,19

3.2 Results

Our proposed model has been used to build a hybrid forecasting system that has been tested along the north west coast of the Iberian Peninsula with data collected by the CCCMM from the year 1992 until the present time. The prototype used in this experiment was set up to forecast the concentration of the pseudo-nitzschia spp diatom of a water mass situated near the coast of Vigo (geographical area A0 ((42°28.90' N, 8°57.80' W) 61 m)), with a week of advance. Although the aim of this experiment is to forecast the value of this concentration, the most important thing is to identify in advance if the concentration will cause a *bloom*.

The average error in the forecast was found to be 26.043,66 cel/liter and only 5.5% of the forecasts had an error higher than 100.000 cel/liter. Although the experiment was carried out using a limited data set, it is believed that these error value results are significant enough to be extrapolated over the whole coast of the Iberian Peninsula.

Two situations of special interest are those corresponding to the *false alarms* and the *not detected blooms*. The first one happens when the model predicts bloom (concentration of pseudo-nitzschia ≥ 100.000 cel/liter) and this doesn't take place (real concentration ≤ 100.000 cel/liter). The second, more important, arise when bloom really exists and the model doesn't detect it.

Table 2 shows the predictions carried out with success (in absolute value and %) and the erroneous predictions differentiating the not detected blooms and the false alarms. This table also shows the average error obtained with all the techniques. As it can be shown, the combination of different techniques in the form of the hybrid CBR system previously presented, produces better results than a RBF neural network working alone or anyone of the tested statistical techniques. This is due to the effectiveness of the revision subsystem and the retrained of the RBF neural network with the cases recovered by GCS network. The hybrid system is more accurate than any of the other techniques studied during this investigation.

4 Conclusions and Future Work

In summary, this paper has presented an automated hybrid CBR system that employs case-based reasoning to wrap a growing cell structures network (for the index tasks to organize and retrieve relevant data), a radial basis function network (that contributes generalization, learning and adaptation capabilities) and a set of Sugeno fuzzy models (acting as experts that revise the initial solution) to provide a more effective prediction. The resulting hybrid system

thus combines complementary properties of both connectionist and symbolic AI methods in order to create a real time autonomous forecasting system.

The developed prototype is able to produce a forecast with an acceptable degree of accuracy. The results obtained may be extrapolated to provide forecasts further ahead using the same technique, and it is believed that successful results may be obtained. However, the further ahead the forecast is made, the less accurate the forecast may be expected to be. The developed prototype can not be used in a particular geographical area if there are no stored cases from that area. Once the system is in operation and it is forecasting, a succession of cases will be generated, enabling the hybrid forecasting mechanism to evolve and to work autonomously.

In conclusion, the hybrid reasoning problem solving approach may be used to forecast in complex situations where the problem is characterized by a lack of knowledge and where there is a high degree of dynamism. The model presented here will be tested in different water masses and a distributed forecasting system will be developed based on the model in order to monitor 500 km. of the North West coast of the Iberian Peninsula.

This work is financed by the project: *Development of techniques for the automatic prediction of the proliferation of red tides in the Galician coasts*, PGIDT-00MAR30104PR, inside the Marine Program of investigation of Xunta de Galicia. The authors want to thank the support lent by this institution, as well as the data facilitated by the CCCMM.

References

- [1] Corchado, J. M., and Lees, B.: Adaptation of Cases for Case-based Forecasting with Neural Network Support. In Pal, S. K., Dillon, T. S., and Yeung, D. S. (Eds.). *Soft Computing in Case Based Reasoning*. London: Springer Verlag, (2000) 293–319
- [2] Pal, S. K., Dillon, T. S., and Yeung, D. S.: *Soft Computing in Case Based Reasoning*. Springer Verlag: London, (2001)
- [3] Corchado, J. M., Lees, B.: A Hybrid Case-based Model for Forecasting. *Applied Artificial Intelligence*, 15, num. 2, (2001) 105–127
- [4] Corchado, J. M., Aiken, J., Rees, N.: *Artificial Intelligence Models for Oceanographic Forecasting*. Plymouth Marine Laboratory, U.K., (2001)
- [5] Fritzke, B.: Growing Self-Organizing Networks-Why?. In Verleysen, M. (Ed.). *European Symposium on Artificial Neural Networks, ESANN-96*. Brussels, (1996) 61–72
- [6] Fritzke, B.: Fast learning with incremental RBF Networks. *Neural Processing Letters*, 1, num. 1, (1994) 2–5
- [7] Jin, Y., Seelen, W. von., and Sendhoff, B.: Extracting Interpretable Fuzzy Rules from RBF Neural Networks. Internal Report IRINI 00-02, Institut für Neuroinformatik, Ruhr-Universität Bochum, Germany, (2000)
- [8] Azuaje, F., Dubitzky, W., Black, N., and Adamson, K.: Discovering Relevance Knowledge in Data: A Growing Cell Structures Approach. *IEEE Transactions on Systems, Man and Cybernetics*, 30, (2000) 448–460
- [9] Jang, J.-S.R., Sun, C.-T.: Functional equivalence between Radial Basis Function Networks and Fuzzy Inference Systems. *IEEE Transactions on Neural Networks*, 4, (1993) 159
- [10] Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15, (1985) 116–132
- [11] Fernández, E.: *Las Mareas Rojas en las Rías Gallegas*. Technical Report, Department of Ecology and Animal Biology. University of Vigo, (1998)

Rule Extraction from Bagged Neural Networks

Guido BOLOGNA

*Swiss Institute of Bioinformatics
 Rue Michel Servet 1 Geneva, CH-1211 Switzerland*

Abstract. This work presents a technique to extract rules from ensembles of *Discretized Interpretable Multi Layer Perceptrons* (DIMLPs) based on the characterization of discriminant hyper-planes. Experiments were achieved on 25 classification problems using single DIMLP networks and bagged ensembles. It turned out that extracted rule sets from bagged DIMLPs were on average significantly more accurate than single networks (78.0% versus 76.4%), and slightly less complex. Finally, rules were slightly more accurate than those generated from ensembles of C4.5 decision trees (78.0% versus 77.8%), while exhibiting significantly smaller complexity in almost all classification problems.

1 Introduction

Explaining neural network responses by symbolic rules given as “*if tests on antecedents are true then conclusion*” is difficult and plays often a significant part to the mysticism related to the use of neural networks. A taxonomy describing techniques used to generate symbolic rules from neural networks was proposed by Andrews *et al.* [1]. Note that other forms of knowledge representation given as finite state automata have been investigated by Giles *et al.* from recurrent neural networks [10].

Basically, rule extraction methods are grouped into three methodologies: *pedagogical*, *decompositional*, and *eclectic*. In the pedagogical methodology, symbolic rules are generated by an inductive learning algorithm applied to the associations between input-output patterns. This is also called the black-box approach. For decompositional techniques, symbolic rules are determined by inspecting the weights at the level of each neuron. Finally, the eclectic methodology both combines elements of the pedagogical and decompositional methodologies.

The majority of rule extraction techniques applied to standard multi-layer perceptrons are decompositional [1]. More precisely, for a given neuron those algorithms approximate the activation function by a step function and then search for combinations of pre-synaptic values activating or deactivating this neuron. In this case the number of possible combinations scales exponentially with the number of pre-synaptic inputs; thus several heuristics must be used to reduce the search space. For instance Saito and Nakano [16] limits the maximal number of antecedents in a rule, Krishnan *et al.* take into account the magnitude of weights and generate rules by first considering the most important weights [12]. If the number of connections is not kept small by pruning techniques [17], several important rules can be missed, as the implicit

search tree is pruned by heuristics. Finally, all compositional methods applied to standard multi-layer perceptrons assume that inputs are binary-valued; thus, inputs when continuous must be converted into binary values.

Examples of pedagogical and eclectic approaches have been proposed by Craven and Shavlik [6] with *Trepan*, and Setiono with *Neuro-Linear* [18]. *Trepan* is a general algorithm that can be applied to any network architecture. *Trepan*'s computational complexity scales in polynomial time with the size of the problem and the size of the network. The eclectic *Neuro-Linear* technique first uses C4.5 decision trees [14] to determine rules related to hidden layer activations and output layer responses. Then, a compositional approach is applied to patterns related to the input layer and the hidden layer. Because weights are pruned during training, the compositional part of the rule extraction technique is performed for many problems in reasonable time, though exponential in its nature. The interested reader will find other rule extraction technique descriptions in [9].

In a *Discretized Interpretable Multi Layer Perceptron* (DIMLP), discriminant frontiers are axis-parallel hyper-planes [3]. Rules are generated by the induction of a special decision tree taking into account virtual hyper-plane frontiers. With respect to the training set, the degree of matching between network responses and extracted rules also denoted as *fidelity* is 100%. Moreover, the computational complexity of the rule extraction algorithm scales in polynomial time with the dimensionality of the problem, the number of examples, and the size of the network. Finally, continuous attributes do not need to be binary transformed as it is done in the majority of rule extraction techniques.

Recently, voting methods based on several combined classifiers have been investigated [13]. Very often the accuracy of an ensemble of classifiers is higher than the accuracy of a single classifier. In order to generate rules from ensembles, Quinlan extracted single C4.5 decision trees to describe ensembles of three decision trees [15]. Likewise, Domingos presented *Combined Multiple Models* (CMM), a meta-learner based on C4.5 decision trees that retains most of the accuracy gains of multiple model approaches and that produced comprehensible rules [8]. The key idea behind CMM was the use of the base model to learn a large number of examples generated and classified according to the ensemble, plus the original examples. Craven applied his general rule extraction technique to ensembles of neural networks to extract decision trees [7]. Finally, the author extended his rule extraction technique to generate rules from ensembles of DIMLP neural networks [3].

Here, we present a comparison study on rules extracted from ensembles of DIMLP neural networks and ensembles of C4.5 decision trees. On 25 classification problems we use ensembles of 25 classifiers trained by *bagging* [5]. To the best of our knowledge, this is the first work that aims at comparing rules generated from ensembles of neural networks and ensembles of decision trees based on a large number of classification tasks. In the remaining sections, section two introduces the DIMLP neural network model, section three presents the rule extraction algorithm, section four explains the learning algorithm, section five describes ensembles of DIMLPs, and section six shows the experiments followed by the conclusion.

2 The DIMLP Model

In the DIMLP model there are an input layer, one or more hidden layers, and an output layer. As an example, figure 1 illustrates a network with two hidden layers. Note that neurons are not fully connected between the input layer and the first hidden layer. Moreover, the activation

function of neurons in the first hidden layer is the staircase activation function instead of the sigmoid function. These two differences with respect to the standard multi-layer perceptron (MLP) will allow us to extract symbolic rules.

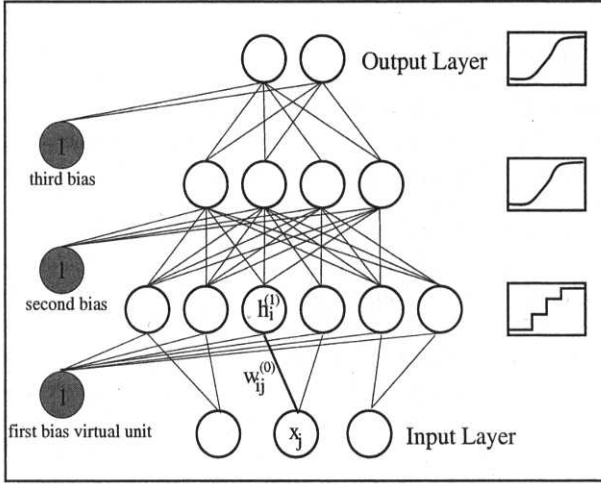


Figure 1: A DIMLP network.

The activation of a neuron that is above the first hidden layer is the sigmoid function given as

$$act(t) = \sigma(t) = \frac{1}{1 + \exp(-t)}. \quad (1)$$

For neurons of the first hidden layer the activation is a staircase activation function that approximates a sigmoid. Note that, several properties characterize the DIMLP model with respect to standard multi layer perceptrons. Here we enumerate three results; more details can be found in [4].

Result 1

In a DIMLP network with one hidden layer or more, the use of the staircase activation function builds a lattice of hyper-rectangles. In a hyper-rectangle the activation of all neurons of the output layer is constant.

Result 2

An axis-parallel hyper-plane between two hyper-rectangles is a possible discriminant frontier. As a definition, we will denote a possible discriminant hyper-plane as a *virtual* hyper-plane.

Result 3

For a staircase activation function with q stairs of length δ , virtual hyper-planes lie in

$$v_k^{(ij)} := \frac{a + \delta k - w_{i0}^{(0)}}{w_{ij}^{(0)}}, \quad (2)$$

where a is a constant depending on a staircase function, $w_{i0}^{(0)}$ is a weight connection between the first bias virtual neuron and the i^{th} neuron of the first hidden layer, $w_{ij}^{(0)}$ is a weight connection between the j^{th} input neuron and the i^{th} neuron of the first hidden layer, and with $0 \leq k \leq q$.

3 Rule Extraction from a DIMLP Network

The number of hyper-rectangles built by a DIMLP network is exponential with respect to the number of input attributes. Generally, the rule extraction task corresponds to the resolution of a covering problem where discriminant hyper-planes represent frontiers between regions of different classes. The purpose of the rule extraction algorithm is to cover with a minimal number of symbolic rules all hyper-rectangles containing training examples. Even when we restrict the covering problem to available examples, the search for the minimal covering is NP-hard. However, sub-optimal solutions can be found in polynomial time by heuristic methods. Among several algorithms, it was decided to use decision trees as an important component of our rule extraction technique, because they are often faster than other algorithms.

Briefly, a decision tree is built by a recursive function splitting the input space by axis-parallel hyper-planes. At each step of the algorithm a criterion to determine the best split is used. This technique corresponds to a “Divide and Conquer” approach. One of the most popular decision tree models is C4.5 [14]. In such decision tree, the inherent learning mechanism is based on a uni-variate search technique, whereas the training phase of a neural network is based on a multi-variate search. Therefore, during the training phase a decision tree may miss rules involving multiple attributes which are weakly predictive separately, but become strongly predictive in combination. On the other hand, a neural network may fail to discern a strongly relevant attribute among several irrelevant ones. Because we aim at using decision trees to extract rules from DIMLP networks, the bias related to the decision tree search was modified by implementing the criterion of the *relevance* of a hyper-plane, instead of the standard gain entropy measure [14].

3.1 The relevance of a hyper-plane

The relevance of a hyper-plane corresponds to the number of cases viewing this hyper-plane as the transition to a different class. Figure 2 illustrates the concept of relevance of hyper-planes. Twelve points belonging to two classes correspond to the classification given by a DIMLP network. Moreover vertical and horizontal lines are virtual hyper-planes. Among all vertical lines, $\{x_1 = t_{13}\}$ is the most relevant, because all white squares on the left see this line as the transition to the class of black circles on the right. Further, all black circles on the left see $\{x_1 = t_{13}\}$ as the transition to the class of white squares on the right. It is worth

noting that this is a situation where each single feature has no discriminant power, but the combination of the two features is highly predictive.

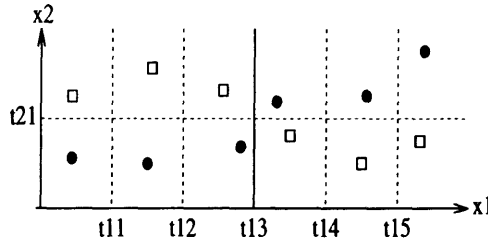


Figure 2: The relevance of a hyper-plane. Among all vertical lines, lines $\{x_1 = t_{13}\}$ is the most relevant, with relevance equal to 12. For all other lines the relevance is equal to zero.

3.2 The Rule Extraction Algorithm

The rule extraction algorithm is:

1. Determine virtual hyper-planes (cf. (2)).
2. Determine the relevance of hyper-planes using training examples and DIMLP classifications.
3. Build a decision tree according to the most relevant hyper-plane.
4. Prune rule antecedents according to the most enlarging criterion.
5. Prune unnecessary rules. Goto 4 if there are other antecedents to prune.
6. Modify antecedent thresholds according to the most enlarging criterion.
7. Prune unnecessary rules. Goto 6 if at least one antecedent has been modified.

In the first step of the rule extraction algorithm, virtual hyper-planes are precisely determined by weight values connecting an input neuron to a hidden neuron (cf. (2)). In the second step, the relevance of discriminant hyper-planes is estimated from all available examples and DIMLP responses. Very often, several hyper-planes are not relevant, and thus are discarded, especially when the grid is very finely grained.

Figure 3 illustrates a rule extraction example for a classification problem with two classes. In A, we show the class of the data samples given by a DIMLP network and the relevance of the hyper-planes at the root of the decision tree. A decision tree is built according to the *most relevant hyper-plane* criterion. Each path between the root and a leaf of the final decision tree corresponds to a rule. At this stage rules are disjointed and reflect neural network classifications on the training set (see figure 3.B). To reduce the number of antecedents, a pruning strategy is applied to all rules according to the *most enlarging antecedent* criterion. More precisely, for all possible antecedents, the antecedent that most increases the number

of covered examples is pruned. Note that, any modification to the current rule set in steps 4, 5, 6, and 7 is carried out if the fidelity of the new rule set is still 100%, with respect to the training set. In figure 3.C, we illustrate step 4. More precisely, for rule R_2 the number of covered examples has increased from 2 to 9 when the second antecedent has been pruned.

When it is no longer possible to prune any antecedent or any rule, again, to increase the number of covered examples by each rule, all thresholds of remaining antecedents are modified according to the *most enlarging* criterion. More precisely, for each attribute new threshold values are determined according to the list of discriminant hyper-planes (cf. step 2). At each step, the new threshold antecedent which most increases the number of covered examples without altering DIMLP classifications is retained. An example of this procedure is shown in figure 3.D. In fact, only the first antecedent of R_4 is able to be changed. Consequently, R_2 is removed, as is contained in R_3 and R_4 .

In the worst case, all steps of the rule extraction algorithm scale in polynomial time with respect to the number of examples, attributes, and stairs. Therefore, this algorithm can be used for reasonably large DIMLP networks and reasonably large problems.

4 Learning in a DIMLP Network

The training phase is carried out by varying the weights in order to minimize the *Mean Squared Error* function [11]. Although this function is not differentiable with staircase activation functions we use an on-line back-propagation algorithm with momentum (μ) and flat spot elimination parameter (FSE). More precisely, during training the gradient is determined in each layer by the use of sigmoid functions, whereas the error related to the stopping criterion is calculated using staircase activation functions in the first hidden layer. The training algorithm is:

1. Compute the output for an example using sigmoid activation functions in the first hidden layer.
2. Compute the gradient of the error.
3. Modify weights according to the gradient of the error.
4. If all examples have been presented compute the mean squared error of all examples using staircase activation functions in the first hidden layer.
5. If stop criterion reached (depending on the mean squared error) goto 1; else stop.

Generally, because the staircase activation function represents a quantization of the sigmoid function, the difference of the responses given by a network with staircase activation functions and a network with sigmoid activation functions tends to zero when the number of stairs is sufficiently large. As an heuristic from experience, 50 stairs are sufficient to learn a large number of data sets.

5 Ensembles of DIMLP Networks

Combining the predictions of several classifiers to produce a single classifier is generally more accurate than any of the individual classifier making up the ensemble. Our purpose is

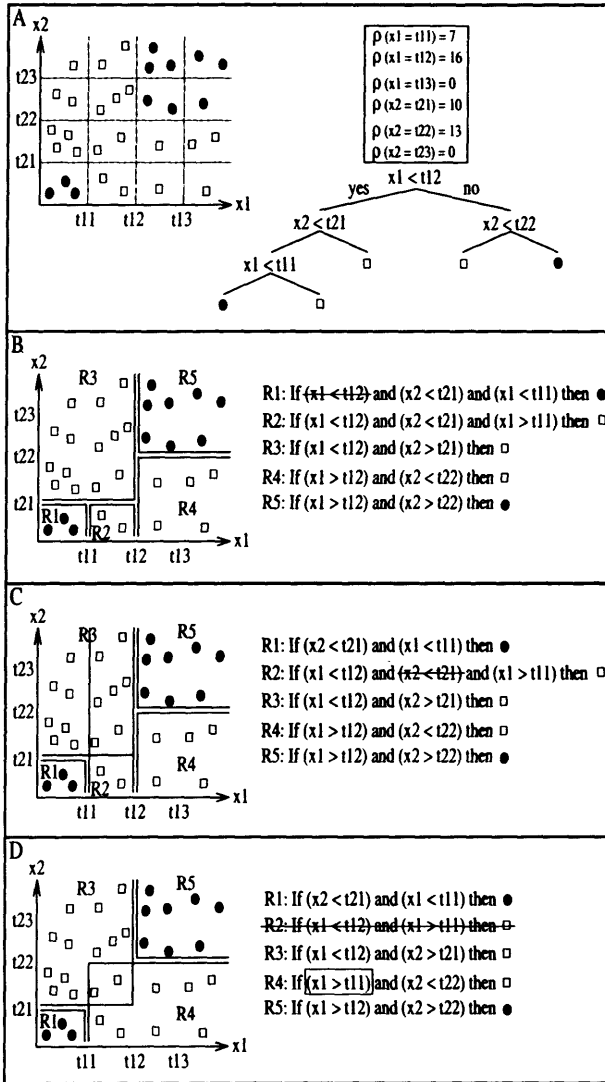


Figure 3: A rule extraction example for a classification problem with two classes. In A, dashed lines represent virtual hyper-planes. A tree is built according to the most relevant hyper plane; here we show the relevance (ρ) for the root of the tree (for the other nodes the relevance is not shown). In B, each path from the root to a leaf of the decision tree is a disjointed rule. Note that in rule R_1 antecedent $(x_1 < t_{12})$ is removed, as $(x_1 < t_{11})$ and $(x_1 < t_{12})$ is equivalent to $(x_1 < t_{11})$. In C, an antecedent of R_2 is pruned (there are no other choices). Finally, in D a threshold of an antecedent is modified and rule R_2 is removed.

to use linear combinations of DIMLPs to improve the predictive accuracy of a single DIMLP network. More precisely, to build these ensembles we adopt the bagging strategy introduced by Breiman [5].

5.1 Bagging

Assuming that P is the size of the original training set, bagging generates for each classifier P examples drawn with replacement from the original training set. As a consequence, for each network many of the generated examples may be repeated while others may be left out. More precisely, for large data sets, the probability a training example will be part of a new resampled training set is approximately 63.2%.

5.2 Rule Extraction

The rule extraction technique presented in section 3 can be applied to any DIMLP architecture having as many hidden layers as desired. In fact, the first hidden layer determines the exact location of hyper-planes, whereas all other layers just switch discriminant frontiers on or off in a given region of the input space (cf. Result 1). Note that the linear combination of several DIMLP networks is again a DIMLP network with special connectivity. More precisely, for two consecutive layers and for two different networks of the ensemble, all weights between the two networks are equal to zero.

6 Experiments

The purpose was to compare rules extracted from ensembles of 25 DIMLP networks and ensembles of 25 decision trees. Twenty-five classification problems were selected from the University of California public domain [2]. Results for bagged decision trees were those reported by Domingos¹ [8]. Table 1 gives the characteristics of the data sets, as well as DIMLP neural architectures.

6.1 Methodology

The number of neurons in the hidden layers was based on the heuristic that the number of weights must be less than the number of examples, and three neurons in the second hidden layer being a minimum. Moreover, the number of neurons in the second hidden layer was at least equal to the number of output neurons. Finally, DIMLP networks were trained with default parameters. In our implementation of back-propagation default parameters are: *learning parameter* = 0.1; *momentum* = 0.6; *flat spot elimination* = 0.01; *number of stairs* = 50.

The experiments were based on the average calculated after ten repetitions of ten-fold cross-validation. Further, the training phase of single DIMLP networks was stopped according to the minimal error measured on an independent validation set. For each cross-validation trial the proportions of training sets, validation sets, and testing sets were 8/10, 1/10 and 1/10, respectively.

¹We discarded the *Solar Flare* problem, because the definition of the classes is unclear.

Table 1: Data sets and DIMLP architectures.

DATA SET	CASES	ATTR.	CLS.	ARCH.
AUDIOLOGY	200	69	24	102-102-24-24
ANNEALING	898	38	5	81-81-5-5
CANCER	286	15	2	50-50-3-2
CREDIT	690	15	2	44-44-6-2
DIABETES	768	8	2	8-8-30-2
ECHOCARD.	131	7	2	7-7-7-2
GLASS	214	9	6	9-9-10-6
HEART-C	303	13	2	22-22-5-2
HEPATITIS	155	19	2	29-29-3-2
HORSE COLIC	368	28	2	61-61-4-2
IRIS	150	4	3	4-4-12-3
LABOR	57	16	2	29-29-3-2
LED	100	7	10	7-7-10-10
LENSES	24	4	3	4-4-3-3
LIVER	345	6	2	6-6-15-2
LUNG	32	56	3	56-56-3-3
LYMPH.	148	19	4	38-38-4-4
POST-OP.	90	8	3	20-20-3-3
PRIM. TUM.	339	18	22	37-37-22-22
PROMOTERS	106	57	2	228-228-3-2
SONAR	208	60	2	60-60-3-2
SOYBEAN	47	35	4	35-35-4-4
VOTING	435	16	2	48-48-3-2
WINE	178	13	3	13-13-5-3
ZOOLOGY	101	17	7	17-17-7-7

Finally, extracted rules were evaluated on the number of extracted antecedents plus the number of consequents per rule set (the default rule is included with 0 antecedents and 1 consequent). Note that this measure is related to the complexity of a classifier. The more complex a classifier is, the larger is the number of extracted antecedents and consequents. Note also that rules extracted from DIMLP networks are “if-then” expressions (unordered), whereas rules generated from C4.5 decision trees are “if-then-else” expressions (ordered).

6.2 Results

Average predictive results of generated rules from single DIMLPs (DIMLP-R), bagged DIMLPs (DIMLP-B-R), and bagged C4.5 decision trees (C4.5-B-R) are shown in table 2. Note that only in four problems out of 25 (ANNEALING, LYMPH., SOYBEAN, and WINE), rules extracted from bagged DIMLPs did not improve accuracy with respect to rules generated from single networks.

The t statistic for testing the null hypothesis that two means are equal was performed for DIMLP-B-R and C4.5-B-R at the rejection level of 5%. If the null hypothesis was rejected, we checked which method presented the best accuracy. In table 2, a “*” designates a DIMLP-B-R win, whereas a “•” represents a C4.5-B-R win. It turned out that rules generated from bagged DIMLPs were significantly more accurate than rules generated from bagged C4.5 decision trees in 9 problems. Conversely, in 6 problems C4.5-B-R were significantly more

accurate. Finally, the average of average predictive accuracies over all classification problems was slightly better for rules extracted from bagged neural networks than rules extracted from bagged decision trees (78.0% versus 77.8%).

In the last row of table 2 are represented in parenthesis the average of the average predictive accuracies of the original models. All the original classifiers were more accurate than their corresponding generated rule sets. Moreover, the average of the average fidelity of the rules generated from single and bagged DIMLPs with respect to the testing sets were 93.5% and 93.6%, respectively. Finally, by considering all testing samples for which rules and networks agreed, the average of the average predictive accuracies increased to 79.1% and 80.9%, respectively. This is better than the average of the average accuracies of the networks (77.6% and 79.8%; respectively).

Table 2: Average predictive accuracies of extracted rules.

DATA SET	DIMLP-R	DIMLP-B-R	C4.5-B-R
AUDIOLOGY •	72.7±2.0	75.7±0.9	77.8±1.5
ANNEALING ★	99.1±0.2	99.1±0.2	96.2±0.5
CANCER ★	71.2±1.9	74.2±0.5	70.2±1.8
CREDIT •	85.0±0.9	86.2±0.4	87.5±1.1
DIABETES ★	75.4±0.9	76.6±0.7	75.5±0.9
ECHOCARD.	65.9±2.4	69.5±2.0	67.7±3.2
GLASS •	62.6±3.1	66.0±1.7	73.1±1.5
HEART-C	79.8±1.4	81.7±1.0	81.8±1.5
HEPATITIS ★	78.3±2.6	80.5±1.4	76.2±2.5
HORSE COLIC •	80.5±1.3	82.7±1.1	86.3±1.6
IRIS	93.1±1.1	93.2±1.4	94.3±1.1
LABOR	81.1±5.5	84.5±3.7	85.8±2.5
LED	58.8±3.6	61.7±1.8	61.5±3.2
LENSES	68.3±9.7	69.5±7.1	75.0±6.8
LIVER ★	68.7±1.4	70.5±1.9	66.0±1.8
LUNG	43.1±7.9	43.8±5.5	40.0±7.5
LYMPH. ★	80.0±2.2	78.7±1.7	76.7±2.3
POST-OP. ★	67.0±4.4	71.1±0.0	68.9±3.0
PRIM. TUM. ★	42.4±1.5	45.4±0.7	43.4±2.1
PROMOTERS •	83.7±4.8	84.2±1.6	87.7±1.7
SONAR ★	78.3±2.7	79.1±2.1	71.7±2.2
SOYBEAN	96.6±2.7	95.4±3.7	97.0±1.6
VOTING	95.0±0.7	95.6±0.6	95.5±0.7
WINE •	91.7±1.6	91.4±2.1	94.2±1.0
ZOOLOGY	92.7±1.9	94.9±1.6	94.0±1.8
AVERAGE	76.4 (77.6)	78.0 (79.8)	77.8 (78.9)

A comparison exhibiting the complexity of rules is shown in table 3. For most of the problems, rule sets extracted from bagged DIMLPs were slightly less complex than rules generated from single networks. Moreover, in many cases, rules extracted from bagged C4.5 decision trees were significantly more complex than rules generated from bagged networks. This is probably related to the fact that in [8] a substantial number of examples is generated to increase the fidelity of rules, while in our case we just use the training set.

Table 3: Complexity of rules.

DATA SET	DIMLP-R	DIMLP-B-R	C4.5-B-R
AUDIOLOGY	200.9	210.5	440.3
ANNEALING	59.0	55.0	90.2
CANCER	35.2	14.7	174.1
CREDIT	97.5	74.6	104.9
DIABETES	180.0	139.0	82.4
ECHOCARD.	19.6	15.5	85.1
GLASS	114.2	94.1	214.6
HEART-C	98.1	94.8	177.7
HEPATITIS	32.5	19.8	158.2
HORSE COLIC	97.3	49.6	52.8
IRIS	20.8	14.0	17.6
LABOR	17.0	15.6	39.7
LED	71.9	72.6	202.1
LENSES	10.4	12.5	17.5
LIVER	148.4	137.8	116.1
LUNG	14.6	23.1	198.8
LYMPH.	58.7	59.6	213.2
POST-OP.	12.2	1	82.4
PRIM. TUM.	323.6	311.6	579.0
PROMOTERS	49.7	61.2	143.0
SONAR	92.9	82.1	90.6
SOYBEAN	10.5	10.2	160.1
VOTING	49.1	37.2	74.2
WINE	43.6	39.6	69.4
ZOOLOGY	24.8	23.7	140.8
AVERAGE	75.3	66.8	149.0

7 Conclusion

We presented an algorithm whose purpose was to extract rules from ensembles of DIMLP networks. The main characteristics of this algorithm are that (1) the computational complexity is polynomial with respect to the size of the problem and the size of the network, (2) fidelity of extracted rules is 100% with respect to the training set, and (3) attributes do not need to be quantized.

We tested ensembles of 25 DIMLP networks on 25 classification problems of the public domain. It turned out that with bagging, extracted rules in terms of complexity were similar to those extracted from single networks, while being significantly more accurate in almost all the problems. Moreover, rules extracted from ensembles of decision trees were significantly less accurate in 9 problems, and significantly more accurate in 6 problems. Finally, over all 25 problems, the average of the average predictive accuracies of rules extracted from bagged classifiers was similar, but rules generated from bagged neural networks presented lower complexity.

References

- [1] R. Andrews, J. Diederich and A.B. Tickle, Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks, *Knowledge-Based Systems*, 8, no. 6, pp. 373–389, 1995.

- [2] C.L. Blake and C.J. Merz, C.J., *UCI Repository of Machine Learning Databases* (<http://www.ics.uci.edu/>), University of California, Department of Information and Computer Science, Irvine CA, 1998.
- [3] G. Bologna, A Study on Rule Extraction from several Combined Neural Networks, *International Journal of Neural Systems*, 11, no. 3, pp. 247–255, 2001.
- [4] G. Bologna, A Model for Single and Multiple Knowledge Based Networks, *Artificial Intelligence in Medicine*, Special Issue on Neural Hybrid Systems, In press.
- [5] L. Breiman, Bagging Predictors. *Machine Learning*, 26, pp. 123–140, 1996.
- [6] M. Craven and J. Shavlik, Using Sampling and Queries to Extract Rules from Trained Neural Networks, *Proceedings of the Eleventh International Conference on Machine Learning*, pp. 37–45, 1994.
- [7] M. Craven, and J. Shavlik, Extracting Tree-Structured Representations of Trained Networks, In Touretzky, Mozer, Hasselmo (Eds.), *Proceedings of Neural Information Systems*, 8, pp. 24–30, 1996.
- [8] P. Domingos, Discovery via Multiple Models, *Intelligent Data Analysis*, 2, no. 3, pp. 187–202, 1998.
- [9] W. Duch, R. Adamczak and K. Grabczewski, A New Methodology of Extraction, Optimization and Application of Crisp and Fuzzy Logical Rules, *IEEE Transactions on Neural Networks*, 12, no. 2, pp. 277–306, 2001.
- [10] C.L. Giles *et al*, Learning and Extracting Finite State Automata with Second-Order Recurrent Neural Networks, *Neural Computation*, 4, no. 3, pp. 393–405, 1992.
- [11] J.A. Hertz, A. Krogh and R.G. Palmer, *Introduction to The Theory of Neural Computation*, Addison-Wesley, 1991.
- [12] R. Krishnan, G. Sivakumar and P. Bhattacharya, A Search Technique for Rule Extraction from Trained Neural Networks, *Pattern Recognition Letters*, 20, pp. 273–280, 1999.
- [13] D. Opitz and R. Maclin, Popular Ensemble Methods: An Empirical Study, *Journal of Artificial Intelligence Research* 11, pp. 169–198, 1999.
- [14] J.R. Quinlan, *C4.5: Programs for Machine Learning*, San Francisco, Morgan Kaufmann, 1993.
- [15] J.R. Quinlan, Mini-Boosting Decision Trees, *Journal of Artificial Intelligence Research*, 1998.
- [16] K. Saito and R. Nakano, Medical Diagnostic System Based on PDP Model, *Proceedings of the International Conference on Neural Networks*, pp. 255–262, 1988.
- [17] R. Setiono and H. Liu, Understanding Neural Networks via Rule Extraction, *Proceedings Int. Joint Conf. Artificial Intell.*, pp. 205–219, 1995.
- [18] R. Setiono, Generating Concise and Accurate Classification Rules for Breast Cancer Diagnosis, *Journal of Artificial Intelligence in Medicine*, 18, pp. 205–219, 2000.

Nonlinear Principal Component Analysis to Preserve the Order of Principal Components

Ryo Saegusa

Shuji Hashimoto

Dept. of Applied Physics, Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555 Japan
E-mail: {ryo, shuji}@shalab.phys.waseda.ac.jp

Abstract

Principal component analysis (PCA) is an effective method of linear dimensional reduction. Because of its simplicity in theory and implementation, it is often used for analysis in various disciplines. However, because of its linearity, PCA is not always suitable, and has redundancy in expressing data. To overcome this problem, some methods of nonlinear PCA have been proposed. However, most of the methods have drawbacks, such that the number of principal components must be predetermined, and also the order of the generated principal components is not explicitly given. In this paper, we propose a hierarchical neural network model composed of a number of multi-layered perceptrons to perform nonlinear PCA that preserves the order of the principal components. Moreover, our method does not need to know the number of the principal components in advance. The effectiveness of the proposed model will be shown through experiments.

1 Introduction

In the field of data analysis, it is important to reduce the dimensionality of data, because it makes easy to understand data and decreases the computational cost. As a method of dimensionality reduction, principal component analysis (PCA) is often used in various areas such as pattern recognition and image processing.

PCA is an orthogonal transformation of the coordinate system where we describe data. A basis of the objective coordinate system efficiently represents data distributed on a linear hyper plane as the coordinate value that are called a principal component. However, when n -dimensional data are distributed on a m ($< n$)-dimensional nonlinear manifold in a n -dimensional Euclidean space, more than m dimensionality is required to describe the data in PCA to make the dimensionality reduction inefficient.

In order to solve these problems, some methods of nonlinear principal component analysis (NLPCA) have been developed [1]-[3]. These methods were classified into the two categories. One is an application of a sandglass-type multi-layered perceptron (MLP) proposed by Irie and Kawato [2]. The other method is based of a partial linear approximation by Hastie et al [3]. In these methods, the number of principal components has to be specified in advance. Unfortunately, these methods do not provide a way for deciding the number of principal components aside from inefficient trial and errors. Another drawback of these methods is, even after the adequate number of principal component is given, they do not provide a parameter that corresponds to eigenvalue in the conventional PCA, consequently the ratio of contribution of the respective principal component cannot be determined explicitly. These drawbacks will limit the usage of the methods in real world problems.

In this paper, we propose a novel method of nonlinear principal component analysis that preserves the order of principal components based on their ratio of contributions. Moreover, our method does not need to know the number of the principal components in advance. In the proposed method, an array of neural networks are trained to build a set of nonlinear function that map an input vector to its corresponding vector in the principal component space. The proposed model also has the ability to reconstruct the high dimensional data from its low dimensional

representation in the principal component space. These functions are automatically adjusted in a training process. We also discuss the property of these functions by analyzing experimental results. The section 2 of this paper describes the formulations of the proposed nonlinear PCA. The section 3 demonstrates the numerical experiments. In the section 4, we shall discuss the property of the proposed method. The conclusion and future works are also given.

2 The formulation of nonlinear PCA

2.1 Extension of PCA to nonlinear PCA

Consider a random variable $\mathbf{x} \in R^n$ of $E[\mathbf{x}] = 0$. In PCA, the feature (principal component) vector $\mathbf{y} \in R^m$ ($m \leq n$) is an orthogonal transformation of data \mathbf{x} , described as,

$$\mathbf{y} = W^T \mathbf{x} \quad (1)$$

$$= (\mathbf{e}_1^T \mathbf{x}, \mathbf{e}_2^T \mathbf{x}, \dots, \mathbf{e}_m^T \mathbf{x})^T, \quad (2)$$

where columns of W are orthonormal bases $\{\mathbf{e}_i\}_{i=1, \dots, m}$ that form a m -dimensional linear subspace L .

The reconstructed vector $\hat{\mathbf{x}} \in R^n$ from \mathbf{y} is given as,

$$\hat{\mathbf{x}} = W \mathbf{y} \quad (3)$$

$$= \sum_{i=1}^m \mathbf{e}_i (\mathbf{e}_i^T \mathbf{x}). \quad (4)$$

$\{\mathbf{e}_i\}_{i=1, \dots, m}$ are decided as to minimize,

$$E = E[\|\mathbf{x} - \hat{\mathbf{x}}\|^2].$$

The minimization is equivalent to the maximization of the variance of \mathbf{y} [1].

In PCA, the mapping from the data space to the feature space is linear. The reverse mapping is also linear. We call the former a linear extraction function and the latter a linear reconstruction function.

Above mentioned conventional PCA shows the best performance when the data \mathbf{x} are distributed on an m -dimensional hyper plane. While, when the data are distributed on an m -dimensional nonlinear manifold embedded in n -dimensional Euclidean space such as a curved hyper surface, the nonlinear PCA to provide nonlinear mapping functions should be introduced for non-redundant dimensionality reduction as to represented the data with curvilinear coordinate system.

We define a nonlinear extraction function from data \mathbf{x} onto the feature vector \mathbf{y} as

$$\mathbf{y} = \phi(\mathbf{x}), \quad \phi \in S_e, \quad (5)$$

and a nonlinear reconstruction function from \mathbf{y} onto the reconstructed vector $\hat{\mathbf{x}}$ as

$$\hat{\mathbf{x}} = \psi(\mathbf{y}), \quad \psi \in S_r, \quad (6)$$

where S_e and S_r are the sets of nonlinear functions. Data are nonlinearly corresponded to principal components through the nonlinear extraction function and the nonlinear reconstruction function.

Our problem is to minimize the mean square reconstruction error:

$$E = E[\|\mathbf{x} - \hat{\mathbf{x}}\|^2] \quad (7)$$

$$= E[\|\mathbf{x} - \psi(\phi(\mathbf{x}))\|^2]. \quad (8)$$

When we find the optimal ψ , and ϕ , we can efficiently describe data with less principal components than that of PCA.

2.2 The order of nonlinear principal components

In PCA, a reconstruction function is a linear combination of the bases. The bases $\{e_i\}_{i=1,\dots,m}$ are obtained from the minimization of the mean square reconstruction error under the constraint that they are normalized $\|e_i\| = 1$.

After the bases were obtained, we generate the reconstruction function from a linear combination of the bases. We can design the number to combine the bases for the purpose such as the high contribution ratio, the low cost of memory and the best recognition rate. Since the bases are combined in order of significance, the description efficiency is maximized for the number of principal components to use.

In our method, we introduce nonlinear reconstruction functions that map a vector in the principal component space into the original input space as follows,

$$\hat{x}_1 = \psi_1(y_1), \quad (9)$$

$$\hat{x}_2 = \psi_2(y_1, y_2), \quad (10)$$

$$\vdots \quad (11)$$

$$\hat{x}_m = \psi_m(y_1, \dots, y_m). \quad (12)$$

where y_i represents the score of the input vector with respect to the i -th principal component, and \hat{x}_i is the n -dimensional vector reconstructed by the i -th reconstruction function utilizing the representation of the original input vector x in the i -dimensional principal component space.

The i -th component of extraction function ϕ_i and the i th reconstruction function ψ_i are paired in the following manner,

$$\psi_1(x) = \psi_1(\phi_1(x)), \quad (13)$$

$$\psi_2(x) = \psi_2(y_1, \phi_2(x)), \quad (14)$$

$$\vdots \quad (15)$$

$$\psi_m(x) = \psi_m(y_1, \dots, y_{m-1}, \phi_m(x)). \quad (16)$$

The functions of each pair are adjusted to minimize the mean square reconstruction error in the above order. Therefore, y_i can be regarded as the i -th significant nonlinear principal component. In order to obtain the extraction functions $\{\phi_i\}_{i=1,\dots,m}$ and the reconstruction functions $\{\psi_i\}_{i=1,\dots,m}$, we propose hierarchical nonlinear principal component network (HNPCN) composed of MLPs that are arranged hierarchically.

2.3 The hierarchical nonlinear principal component network

The architectures of most sandglass-type MLPs for NLPCA is composed of five layers as shown in Figure 1. The first and the fifth layers are the input and output layers, respectively. It is expected that principal components will be extracted in the third layer, provided that the number of the units in the third layer is less than the number of units in the first layer. The part from the first layer to the third layer has a role of data extraction, while the part from the third layer to the fifth layer has a role of data reconstruction. It should be noted that the number of principal component, which is the number of the units in the third layer, must be determined before the training and there are no differences of significance among the third layer units in this type of NLPCA.

We propose a hierarchical nonlinear principal component network (HNPCN) composed of a number of independent sub-network that can extract ordered nonlinear principal components as explained in the previous section. The number of sub-networks corresponds to the number of the principal components to be extracted. The number of layers in each sub-network is larger than five, and the number of input and output units are equally set to the size of input vector's dimension, while the number of units in the middle layer (extraction layer) corresponds to the index of principal component extracted by the corresponding sub-network. The structure of the model is shown in Figure 2.

The activation function of the units in the input, output, and extraction layers is,

$$f(u) = u, \quad (17)$$

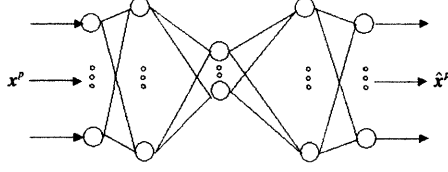


Figure 1: a sandglass-type MLP

while the activation function of the units in the other layer is,

$$f(u) = \frac{1}{1 + \exp(-\frac{u}{T})}. \quad (18)$$

Each sub-network is trained to reconstruct an input vector \mathbf{x} in its output layer by producing $\hat{\mathbf{x}}$. As we want to extract the first principal component in the first sub-network, we set one unit in its extraction layer. The output of this unit is the representation point in the first principal component of that input. The extraction layer of the second sub-network receives the first principal component of the input vector computed in the first sub-network and independently learn to generate a function that map the input layer into the second principal component. This action is executed hierarchically in all of the sub-networks. It is obvious that the proposed model executes the nonlinear principal component mapping explained in the previous section.

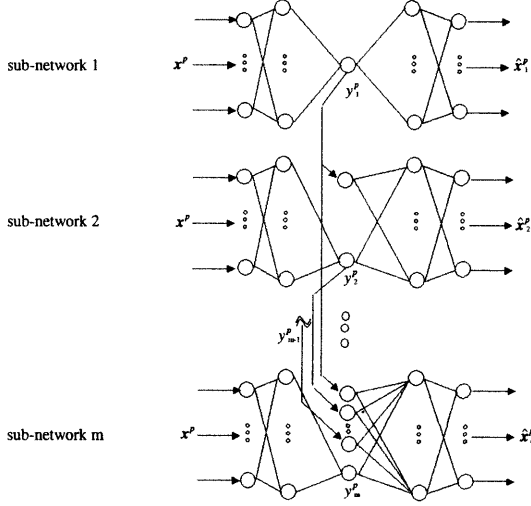


Figure 2: structure of HNPCN

When n -dimensional data \mathbf{x}^p numbered p is given to the first layer of the i -th sub-network, the i -th unit in the extraction layer outputs one-dimensional data

$$y_i^p = \phi_i(\mathbf{x}^p) \in R^1, \quad (19)$$

where the function ϕ_i is the i -th component of the extraction function from \mathbf{x}^p onto a principal component y_i^p . ϕ_i is corresponded to the extraction part that is from all units in the input layer to the i -th unit of the extraction layer, while the principal components $y_1^p, y_2^p, \dots, y_{i-1}^p$ from all upper sub-networks are fed to this layer.

The function ψ_i is the i -th reconstruction function from principal components $y_1^p, y_2^p, \dots, y_i^p$ onto the reconstructed data \hat{x}_i^p . ψ_i is corresponded to the reconstruction part that is from all units of the extraction layer to all units of the output layer.

The calculation of all outputs mentioned above is carried out in the increasing order of the sub-network number. After that, each connection weight of the i th sub-network is adjusted in order to obtain an identical mapping by the criterion of the mean square error

$$E_i^p = \|\mathbf{x}^p - \hat{\mathbf{x}}_i^p\|^2. \quad (20)$$

As to the input data \mathbf{x}^p , the additional correction Δw_i^p of a weight coefficient w_i^p is

$$\Delta w_i^p = -\eta \frac{\partial E_i^p}{\partial w_i^p}, \quad (21)$$

that is adjusted by back propagation algorithm. η is a constant learning parameter.

The w_i^p of the i th sub-network is adjusted only inside the i th sub-network. Error signals of the i th sub-network is not propagated into the reconstruction part of all upper sub-networks. As a result, adjustment of the weight coefficients in the i th sub-network is dependent from the upper sub-networks to the lower ones. It is expected that this dependency forces to construct the lower extraction function both to be different from any of upper $1, 2, \dots, (i-1)$ th functions and to be efficient to reconstruct data.

3 Numerical Experiments

We conducted some experiments to examine the efficiency of the proposed method. In this section we present two experimental results. The one is three-dimensional artificial data, which is easy to understand the distribution, and the other is high dimensional data corresponds to wave form.

3.1 Experiment with three-dimensional artificial data

Training data and test data are the coordinates of points on a parabolic surface in a three-dimensional Euclidean space given as follows,

$$x_3 = \frac{x_1^2}{a_1^2} + \frac{x_2^2}{a_2^2}, \quad (22)$$

in which $(x_1, x_2, x_3) \in R^3$ and $a_1 = 1.0$, $a_2 = 3.0$.

The network in this experiment has two sub-networks. Each sub-network has five layers, where number of units in the first and fifth layer of each sub-network is three, and the number in the second and fourth layer is ten. The parameters for the network are set to $\eta = 0.05$ and $T = 0.1$. The number of training data is 20000, randomly chosen on the parabolic surface. The number of test data is 400 from lattice points on the surface as shown in Figure 3.

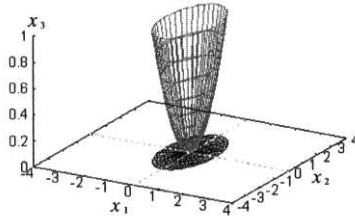


Figure 3: test data

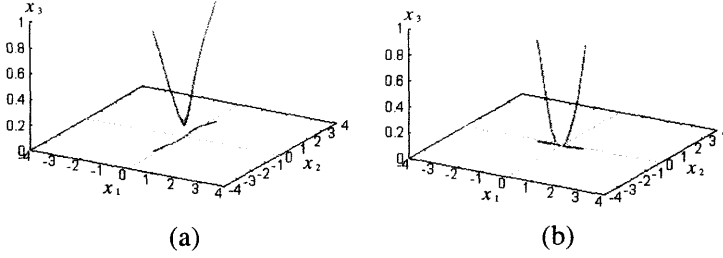


Figure 4: the reconstructed data from the first principal component (a) and the second principal component (b).

Figure 4(a) shows the first principal component, generated from the first sub-network while Figure 4(b) shows the second principal component generated from the second sub-network, without using the output of the unit in the third layer of the first sub-network.

From Eq.(22), the distribution of the data has the greatest variance along

$$x_3 = \frac{x_2^2}{3^2}, \quad (23)$$

which corresponds to Figure 4(a) and the second greatest variance is along

$$x_3 = x_1^2 \quad (24)$$

which is finely reflected in Figure 4(b).

Figure 5 shows the reconstruction of the data from the second sub-network.

It is obvious from this experiment that, the proposed model is able to extract the non-linear principal components from the non-linearly distributed data.

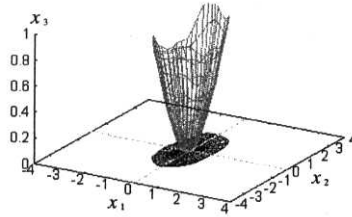


Figure 5: reconstructed data by the second sub-network

Figure 6 shows the principal component scores of the inputs. The horizontal axis indicate the first principal component score, while the vertical axis is for the second principal component score.

We also conducted the experiments with above parabolic surface data embedded in four, five, and six dimensional space. In these high-dimensional cases, the network could almost reconstruct input data from two principal components as well as three-dimensional case.

3.2 Experiment with wave form data

In this experiment the input is a n dimensional vectors, whose components are periodically sampled points of function that is a superposition of three sinusoidal functions. The input vector can be written as follows,

$$(x_1, x_2, \dots, x_n) = (f(\tau + \theta), f(2\tau + \theta), \dots, f(n\tau + \theta)), \quad (25)$$

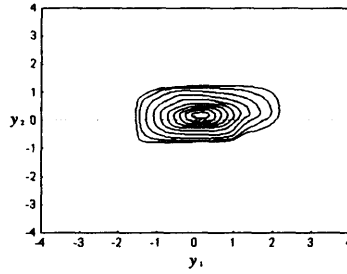


Figure 6: a distribution of the first and second principal component

$$f(s\tau + \theta) = \sum_{k=1}^3 a_k \cdot \sin(\omega_k(s\tau + \theta)) \quad s = 1, 2, \dots, 100, \quad (26)$$

where the frequencies $(\omega_1, \omega_2, \omega_3) = (1.0, 2.0, 3.0)$, the amplitudes $(a_1, a_2, a_3) = (0.5, 0.3, 0.2)$, and $\tau = \frac{2\pi}{100}$. The initial phases θ of the training data are at random. The number of training data is 50000. The number of test data is 100. The number of sub-networks is five and the number of layers is seven through this experiment. We enhanced the description ability of the network by increasing one layer in the extraction part and the reconstruction part in each sub-network. The number of units in the first and seventh layer is 100, and the number in the second, third, fifth and sixth layer is 200. The parameters are set to $\eta = 0.001$ and $T = 0.1$.

In Figure 7 and Figure 8 we show the reconstructed data as to the initial phase $\theta = 0$ and π . The data are reconstructed from test data by the trained network.

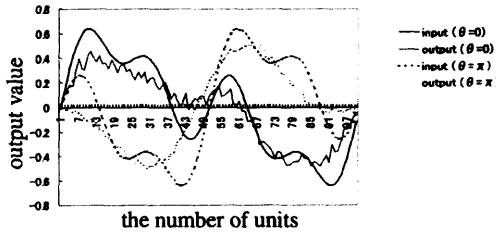


Figure 7: output wave of the first sub-network

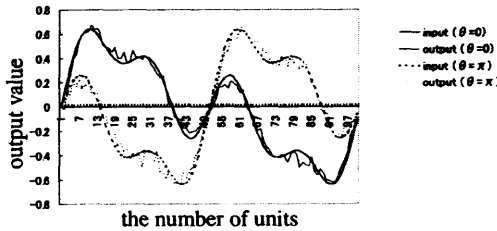


Figure 8: output wave of the fifth sub-network

It is clear from Figure 7 and Figure 8, that the reconstructed wave of the fifth sub-network

that used 5 principal components, is better than that of the first sub-network that used only one principal component. The results verify that although we do not have parameters that correspondent to eigen value like in the conventional PCA, each sub-network is able to extract one principal component, while keeping their order. In Figure 7, the reconstructed data of the first sub-network is similar to a sinusoidal wave. The first extraction function seems to obtain the lowest-frequency sinusoidal function of f . The first extraction function works most efficiently to reconstruct objective wave form because the sinusoidal function has the biggest amplitude.

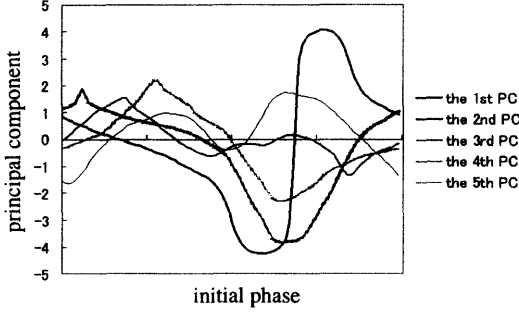


Figure 9: the principal components as to the initial phases

Figure 9 shows the form of the respective principal components. The figure shows the first principal component is a low frequency component, and the frequency gradually grows bigger in the latter principal components. This result is comparable to the Fourier Series, but the proposed model has more independence of expressing the principal components, so that it can be expected that the proposed method will have more efficient principal component representation of some signals that cannot be expressed efficiently by Fourier Series, i.e. signals with discontinuity.

4 Discussions and conclusions

We proposed the method of nonlinear principal component analysis to preserve the order of principal components with the hierarchical neural network model composed of a number of MLPs. In some numerical experiments, we demonstrated that the proposed network constructs the extraction functions in order of the reconstruction efficiency as to objective data. We also confirmed that the network obtains the effective and significant functions for high-dimensional data.

The proposed model has high independence in generating function (extraction function) to map input vectors into their corresponding vectors in the principal component space, while at the same time the model is also able to build a reconstruction function that map the points in the principal component space into the original input space. The forms of the extraction and the reconstruction functions are decided by a number of factors, such as, the structure of the sub-networks, the initial condition of each sub-network and the learning algorithm.

We also examined the robustness of the proposed model with respect to the initial condition of the sub-networks with different structures. This is done with the same data as in the section 3.1.

Figure 10 shows learning MSE of the model, regarding sub-networks with different number of units in the second and the fourth layers. Sub-networks with greater number of units in the second and the fourth layers, have better mapping abilities.

We also examined the performance of the proposed model by setting different numbers of units for the second and fourth layers. In Figure 11(a) the number of the unit in the fourth layer is fixed to 15, while the number of the units in the second layer varies. Oppositely, in Figure 11(b), the number of the units in the second layer is fixed to 15, while the number of units in the fourth layer varies. From Figure 11, we can draw a conclusion that the extraction function

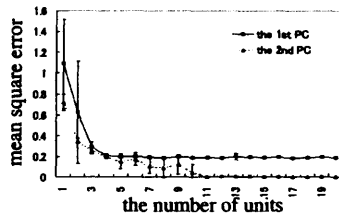


Figure 10: the mean square errors with the various number of units in the second and fourth layer

is valued more than the reconstruction function in this mode. The reason is that a “weak” reconstruction function will fail to make good representation of input vectors in the principal component space, so the reconstructed vectors will not be precise even if the reconstruction function’s performance is strong.

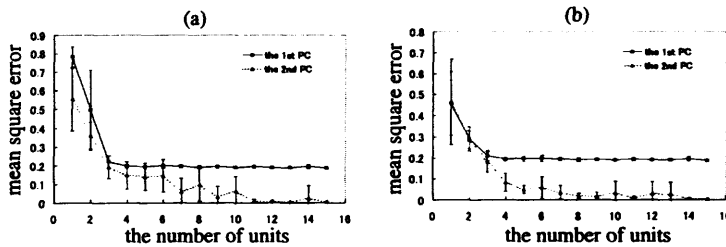


Figure 11: the mean square errors with the various number of units in the second layer under fixing the number in the fourth layer to 15 (a), and the mean square errors with the various number of units in the fourth layer under fixing the number in the second layer to 15 (b).

Although we do not have any explicit parameter that corresponds to the ratio of contributions as in the conventional PCA, the learning error of each sub-network can be used loosely as a guidance in deciding the cumulative ratio of contribution from all the sub-networks.

The promising applications that we consider in the near future encompassed nonlinear pattern classifications, nonlinear data compressions etc.

Acknowledgment

We would like to thank Mr. Hitoshi Sakano of NTT DATA Co. Ltd. for his valuable advices.

References

- [1] K. I. Diamantaras, S. Y. Kung, *Principal Component Neural Networks Theory and Applications*, John Wiley & Sons Inc, 1996.
- [2] B. Irie, M. Kawato, Acquisition of Internal Representation by Multi-Layered Perceptrons, *Journal of IEICE*, Vol.J73-D2 No.8, pp.1173-1178, 1990 (in Japanese).
- [3] T. Hastie, W. Stuetzle, Principal curves, *Journal of the American Statistical Association*, vol.84, no.406, pp.502-516, 1989.

- [4] R. Gnanadesikan, *Methods for statistical data Analysis of multivariate observations*, John Wiley & Sons Inc, 1977.
- [5] R. Saegusa, S. Hashimoto, *Nonlinear principal component analysis using neural networks*, Proc. of the 64th IPSJ general conference, vol.2, pp.205-206, 2002 (in Japanese).
- [6] R. Duda, P. Hart, *Pattern classification theory and systems*, Springer-Verlag, 1988.
- [7] J. Karhunen, J. Joutsensalo, *Generalization of principal component analysis, optimization problems, and neural network*, Neural Networks, Vol.8, No.4, pp.549-562, 1995.
- [8] T. Sanger, *Optimal unsupervised learning in a single-layer linear feedforward neural network*, Neural networks, Vol.2, pp.459-473, 1989.

A Neural Network Model of Rule-guided Behavior

Tetsuto Minami* and Toshio Inui†

Department of Intelligence Science and Technology

Graduate School of Informatics

Kyoto University

Yoshida-honmachi, Sakyo-ku, Kyoto, 606-8501 Japan

Abstract. The flexibility of our behavior is mainly caused by our ability to abstract rules from a circumstance and apply them to other situations. To examine the system for such rule-guided behavior, we proposed a neural network model of rule-guided behavior and simulated the physiological experiments of a rule-guided delayed matching-to-sample task (Wallis et al., 2001). Our model was constructed through neural system identification (Zipser, 1993) and a fully recurrent neural network model was optimized to perform a rule-guided delayed task. In the model's hidden layer, rule-selective units as in Wallis et al. (2001) were found, and an examination of connection weights substantiated that rule-selective neurons maintain encoded rule information and indirectly contributed to rule-guided responses. The simulation results predict functional interactions among neurons exhibiting various task-related activities.

1 Introduction

Much attention has been directed to 'rules' that guide goal-directed behavior. The behavior-guiding rules decide which of different possible outputs to a given input will be produced when more than one response is possible. Rule learning depends upon forming associations between disparate, but behaviorally related, information.

Passingham (1993) and Wise et al. (1996) argue that rule learning is the main function of prefrontal (PF) cortex. The PF cortex provides an ideal infrastructure for playing a role in such learning. Many PF areas receive converging inputs from at least two sensory modalities, and there are ample interconnections between different PF areas. The PF cortex may carry out selection and integration of the information input from sensory areas. Several studies have examined the role of the PF cortex in processing many different types of information: an object and its location (Rao et al., 1997, Rainer et al., 1998), visual and auditory stimuli (Fuster et al., 2000) and 'internal' factors such as attentional state (Rainer et al., 1998). Behavior-guiding rules are a more abstract aspect of information processing in the PF cortex.

*This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for JSPS Fellows, 14002026, 2002.

†This research was supported by the "Research for the Future Program," administered by the Japan Society for the Promotion of Science (Project No. JSPS-RFTF99P01401) and the Project on Neuroinformatics Research in Vision through Special Coordination Funds for Promoting Science and Technology from the Ministry of Education, Culture, Sports, Science and Technology, the Japanese Government.

Humans with PF cortex damage show stereotypical deficits in the WCST (Wisconsin Card Sorting Test), in which subjects are instructed to sort cards according to the shape, color, or number of symbols appearing on them, and the sorting rule varies periodically (Milner, 1963). Wallis et al.(2001) explored the neural basis of rule-guided behaviors by recording single neurons in the PF cortex of monkeys trained to use two rules: a 'match' rule and a 'non-match' rule. The 'match' rule required each monkey to release a lever if two successive sample objects were identical, whereas the 'non-match' rule required release if the two objects were different. The rule applicable for each trial was indicated by a cue that was presented with the sample object. As a result, they found rule-selective neurons that exhibited greater activity when a rule was cued. Other studies have yielded corresponding results (White and Wise, 1999; Hoshi et al.; 2000; Asaad et al., 2000).

Thus, several reports have shown that the principal function of PF cortex involves the application of behavior-guiding rules (White and Wise, 1999; Wallis et al., 2001). However, the mechanism of rule-guided behaviour is still unknown, and the functional role of rule-selective neurons has not been elucidated.

The purpose of this report was to study the mechanisms of rule-dependent neuronal activity in the PF cortex area. Especially, we were interested in how rule-selective neurons contributed to rule-guided behaviors. Based on experimental tasks, we used recurrent neural networks to simulate PF cortex neurons through an approach called neural identification (Zipser, 1992). We simulated the physiological results shown by Wallis et al.(2001), analyzed the temporal patterns of the model units and connection weights, and compared the properties of the units with those of biological neurons.

2 Modeling Methods

The model suggested here was generated using a neural system identification technique. Neural system identification is interesting as a modeling paradigm because of the empirical finding that the internal behavior of neural identification models often mimics the internal behavior of the neural plant being modeled (Zipser, 1992).

The models use simple model neurons with a logistic function. Logistic units represent the average spiking rate of real neurons with a continuous variable, and the synaptic input as a weighted sum of the activations of other neurons. Only the activation of some of the neurons represent model output, while other neurons, hidden units, take part only in intermediate computation steps. These weights are generally chosen automatically by a certain optimization procedure. The optimization procedure is simply a mathematical technique to choose a best set of weights, and is not likely to mimic biological processes. A lot of research has been done using this neural identification paradigm (Zipser and Andersen, 1988; Zipser, 1991; Zipser et al.,1993; Moody et al, 1998, 2000; Xing and Andersen, 2000; Mitchell and Zipser, 2001).

We proposed a neural network model of rule-guided behavior and performed a simulation of the physiological experiment results of Wallis et al. (2001). The model is a fully recurrent neural network model whose neurons are mutually connected.

The model architecture is shown in Fig. 1. In our model, there are three model layers: an input, a hidden, and an output layer. We have already proposed a neural network model of working memory in order to shed light on its retention mechanism and simulated some physiological results shown by Rao et al.(1997) (Minami and Inui, 2001). However, previous

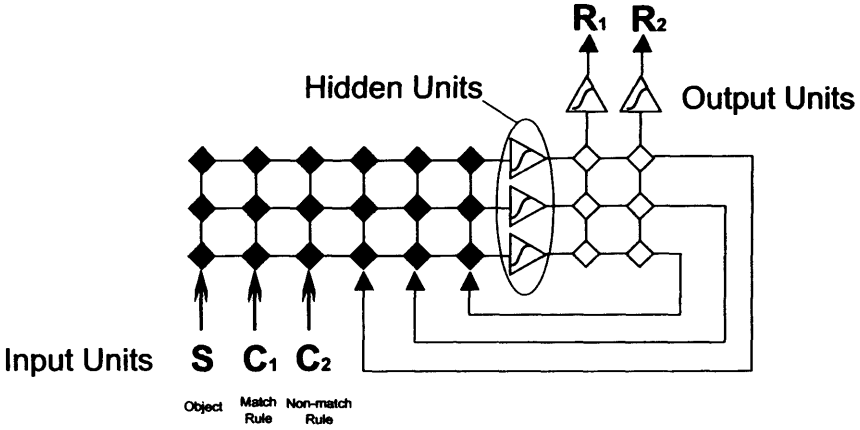


Figure 1: The neural network architecture includes an input, a hidden and an output layer. The input layer consists of an object input and two cue inputs. The hidden layer consists of recurrently connected logistic units. For clarity, only three hidden units are shown here; actual network models contained between 20 and 40 of these units. The diamonds represent the weight of connections

models had comparator units outside of the hidden layer. The present model integrated this function into a hidden layer.

The input layer has three input lines: one for an input which takes a continuous value coding object information, and two for cue information indicating two rules. The cue inputs take a value of (1, 0) when a 'match' rule is applied, and (0, 1) when a 'non-match' rule is applied. The input lines connect to all model units except the output units. Network outputs consist of two units representing 'response': the output (1, 0) means 'release a lever' and (0, 1) 'keep on holding a lever (no response)' in the physiological experiments. The output units receive input from all of the hidden units, but do not feed back to them. The hidden layer consists of recurrently connected logistic units.

The output of the i th model unit in the hidden layer on a time cycle $t + 1$ is given by

$$h_i(t + 1) = f\left(\sum_j h w_{ij} h_j(t) + \sum_k v_{ik} z_k(t) - b_i\right) \quad (1)$$

where $h w_{ij}$ represents recurrent connections from every hidden unit and v_{ik} weighted connections from every input unit z_k . Each unit has a bias, b_i . $f(x)$ is the logistic function

$$f(x) = \frac{1}{1 + e^{-x}} \quad (2)$$

Each motor output on a time cycle $t + 2$ is defined as

$$o_m(t + 2) = f\left(\sum_i o w_{mi} h_i(t + 1) - b_m\right) \quad (3)$$

where $o w_{mi}$ represents the weighted connections from the hidden units. Each unit has a bias, b_m .

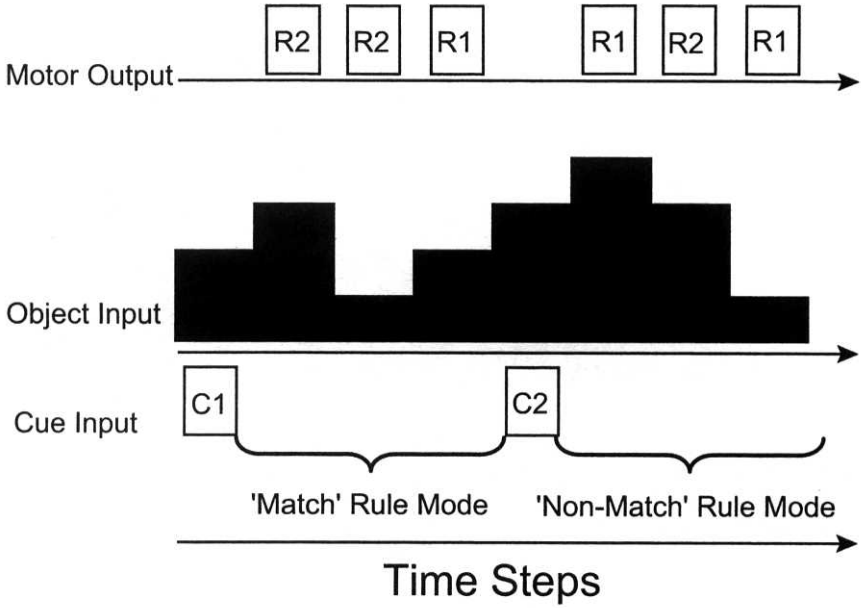


Figure 2: Schematic of training algorithm used for the network. The bottom represents the series of cue inputs and the middle shows those of sample object inputs. The top indicates shows the desired outputs.

The connections were optimized through a gradient descent using RTRL (real-time recurrent learning) (Williams and Zipser, 1989). We used this algorithm to train the network to perform the required behavior with no intention of claiming that the algorithm is similar to the learning mechanisms in the brain. Models were generated by training networks for roughly 10^7 network time steps. Training ceased when the mean square error was less than .01.

To train the model, randomized sequences of object and cue inputs were provided at appropriate times. While training, we used a total of four stimuli ($S_1:0.25$, $S_2:0.5$, $S_3:0.75$, $S_4:1.0$) for object inputs. In the 'match' rule mode, the output R_1 was trained to produce a response when the current object input matched the object input presented when a cue was input, whereas in the 'non-match' rule mode, the output R_2 was trained to produce a response when the current object input did not match the object input presented when a cue was input. Figure 2 provides details of the training paradigm.

3 Results

The model performed this task with unlearned inputs (eg. 0.3, 0.4, 0.6, 0.7), thus showing that it had learned not only simple associations between inputs, but two general rules that could be applied to object inputs.

To determine the mechanism of the model, several analytical approaches were used, based on the following points: the activity pattern during each trial period, the synaptic weight connections and comparisons with physiological data.

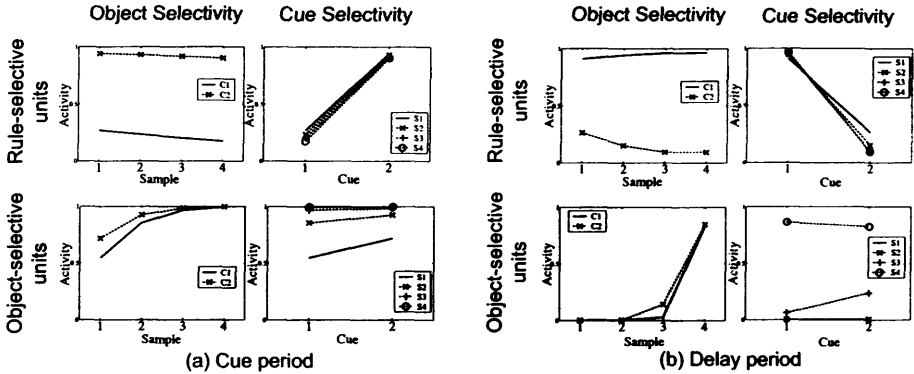


Figure 3: A typical tuning curve to cues and objects during the cue period and delay period: a rule-selective unit and an object-selective unit. Activity plotted as a function of cue (C_1 , C_2) and sample object input (S_1 , S_2 , S_3 , S_4). (a) shows object and cue selectivity during the cue period and (b) shows object and cue selectivity during the delay period.

3.1 Activity patterns during each trial period

In a typical simulated behavioral trial, cue and sample object information were input at the 5th step, corresponding to a cue period. After a delay period (from the 6th to the 9th step), an object was input at the 10th step, corresponding to a response period. We investigated the temporal pattern of hidden units of a typical model network consisting of 30 hidden units. The spectrum of hidden unit activity patterns sustained activity during the delay, or elevated activity only at the time of the inputs. We examined the selectivity of the units during each period: the cue period, delay period and response period.

During the cue and delay periods, rule and object selectivity were observed for many hidden units. We termed these units rule-selective units and object-selective units, respectively. Fig. 3 shows a cue-tuning curve and an object-tuning curve during the cue and delay periods of a rule-selective unit and an object-selective unit. The rule-selective units showed greater response to C_1 indicating the 'match' rule, but had little selectivity to sample objects. On the other hand, the object-selective units showed selectivity to stored object information, regardless of which cue was input.

However, there are mixed type units which show selectivity to both cue and object information. These mixed types have activity profiles that seem to combine rule and object selectivity to various degrees. Such units are more likely to be observed in smaller networks (eg. a network consisting of 20 units).

Fig. 4 shows typical temporal patterns of the rule-selective units. The rule-selective units showed greater activity during match trials, regardless of which object was remembered. The units exhibited the rule selectivity shown in Wallis et al.(2001). The units can contribute to the coding of two learned rules, and object-selective units have selectivity to the stored sample object, which is similar to the object-tuned units of Minami and Inui (2000).

Also, during the response period, many units had selectivity to rule and object information. However, some units showed selectivity not only to inputs, but also to outputs. We termed these units response-related units. R_1 response units show greater activity when R_1

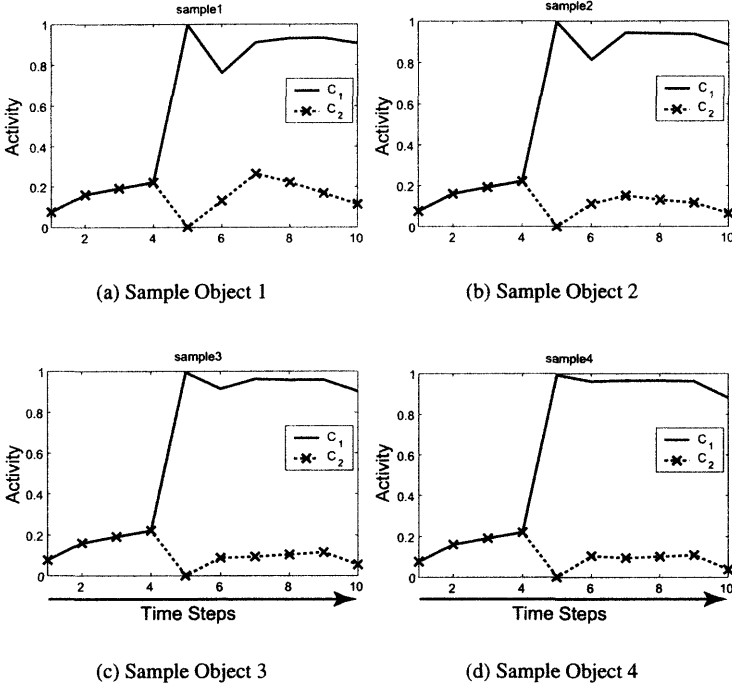


Figure 4: Typical temporal behaviour of the rule-selective unit. The graph shows the temporal pattern when each sample object ($S_1:0.25, S_2:0.5, S_3:0.75, S_4:1.0$) and each cue (C_1, C_2) was input at the 5th step.

output is produced and R_2 response units show greater activity when R_2 output is produced. In the response units, amplitudes of response to current stimuli are modulated by the stored sample object. This modulation includes both potentiation and suppression. An example of how these units respond to all current stimuli for various stored sample object information is shown in Figure. 5. The properties of these response units are very similar to those of the 'comparator' units shown in Moody et al. (1998), but response units have dual-mode activities ('match' rule modes and 'non-match' rule modes) as shown Figure. 5. The shape and peak of the tuning curve remain fixed for different stored objects. The magnitude of the curve is dependent on the current stored objects.

3.2 Synaptic weight connections

Next, we investigated the synaptic weight connections of the model. An examination of the weights between the input layer and the hidden layer showed that rule-selective units had stronger connections with cue inputs, whereas they had weak connections with object inputs. Moreover, the unit that preferred the 'match' rule had positive connections with C_1 and negative connections with C_2 . The units that preferred the 'non-match' rule had the opposite property. This shows that the rule-selective units contribute to the coding of rule information.

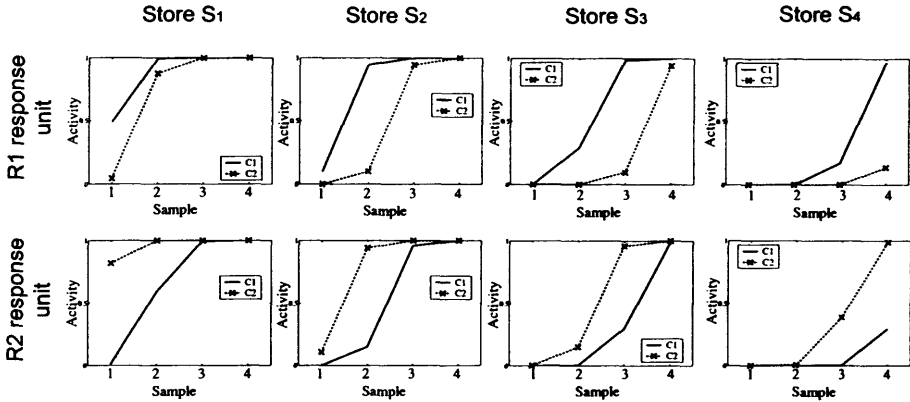


Figure 5: Response unit activity during storage of different sample objects. Activity plotted as a function of current object input. Each plot shows same model unit during storage of four different sample objects ($S_1:0.25$, $S_2:0.5$, $S_3:0.75$, $S_4:1.0$).

Object-selective units had relatively large connective weights with object inputs.

As shown by the selectivity during the response period, large weights that connected response units to the outputs were recognized through the examination of weights between the hidden layer and the output layer. R_1 response units had large positive weights with an R_1 output unit and large negative weights with an R_2 output. That is, R_1 response units facilitate the response of R_1 and suppress that of R_2 , and it is the other way around for R_2 response units. On the other hand, most rule-selective and object-selective units had weaker direct influences on the output units.

Object-selective units received inputs from an object input, had a strong recursive connection, and were mutually connected. In the same way, each rule-selective unit had a strong positive connection with itself. The 'match' rule-selective units had negative weights with the 'non-match' rule-selective units and vice versa. These findings show that rule-selective units maintain rule information using recurrent connections between rule-selective units. The connection weight from rule-selective units to response units was relatively large, but the converse connections were very weak, which shows response units receive rule information from rule-selective units and use this information to judge which mode ('match' rule mode or 'non-match' rule mode) should be applied. Furthermore, response units receive inputs from both the current stimulus and the retained object units. The two types of response units had little interaction with each other.

These analyses of connection weights in the network model can elucidate the mechanism for performing a rule-guided behavior. The network computed output responses via the following method: (1) A rule-selective unit receives input from a cue input, encoding and maintaining rule information. (2) An object-selective unit receives input from an object input and maintains object information. (3) A response unit receives a stored rule and object information from rule-selective units and object-selective units, compares them with current inputs, and produces output signals (Fig. 6). These results suggest that a computational solution in our model may be implemented in a network that includes PF cortex neurons.

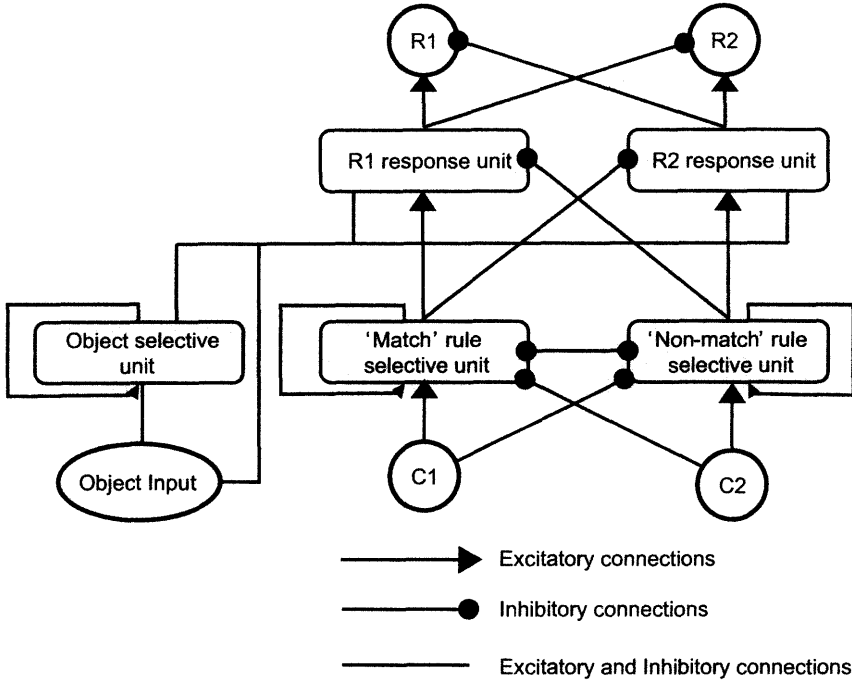


Figure 6: Schematic of the network mechanism: the relationship among the input layer, hidden units and the output layer.

3.3 Comparisons with physiological data

The units of the hidden layer in our model are assumed to correspond to neural networks centering around the PF cortex in the brain. We compared the properties of the model units with those of PF cortex neurons in the physiology experiment results shown by Wallis et al.(2001). Our results confirm that rule-selective neurons in the PF cortex encode and maintain rule information throughout the simulation.

In the physiological experiment, they did not examine the neuronal activities during the response period. Our results predict that some response-related neurons will be found in the brain system centering around PF cortex if the neuronal activities during the response period are investigated.

Funahashi and Inoue(2000) examined functional interactions among PF neurons during a simple oculomotor delayed response task by cross-correlation analysis. They showed a diagram of potential neuronal circuitry in the PF cortex necessary to perform the task (Fig.13 in Funahashi and Inoue, 2000). Our results also predict that similar diagrams to Fig. 6 can be obtained through cross-correlation analysis of neuronal activities during rule-guided behavior.

4 Discussion

Many studies have been done to investigate rule-guided behavior: neuropsychological and neurophysiological studies have shown that PF cortex plays a major functional role in such behavior. However, the mechanisms for rule-guided behavior still remain unclear to neurophysiologists. In this study, we investigated rule-guided behaviour, proposed a neural network model in order to shed light on its mechanism, and simulated some results of physiological experiments concerning the coding of rules.

The main feature of our model is that it is a recurrent neural network, which is interesting for various reasons. A biological neural network has highly recurrent connections. One characteristic of a fully recurrent network is that it can retain information for some time using a fixed point attractor. Zipser(1991) showed that physiological results of neuronal activities of the PF and temporal cortex during a working memory task can be fully explained using the dynamics of this recurrent network.

The model described here augments previous models in several ways. Although active storage has been modeled (Zipser, 1991) and a matching function incorporated (Moody et al, 1998), it has not been previously combined with a rule guided function. Also, our model did not assume a gate or load input with an ambiguous source. The representation of output units is directly linked with motor control and is plausible.

Our model produced the correct outputs for unlearned inputs, thus showing that it had learned two general rules that could be applied to inputs. We classified the hidden units into three classes: rule-selective units, object-selective units and response-related units. Rule-selective units had the same properties as the rule-selective neurons shown in Wallis et al.(2001). Through analysis of the connection weights, we shed light on the mechanism of the PF cortex performing a rule-guided delayed matching-to-sample task, and elucidated the functional role of the rule-selective neurons. The result that the rule-selective units had stronger connections with cue inputs and were mutually connected indicates that the units contribute to the coding and retention of rule information.

One prediction of the model is that there are neuronal interactions in the PF cortex, similar to the diagrams in Fig. 6. In our simulation, we hypothesized that the hidden layer was a neural network centering around the PF cortex. This hypothesis is consistent with recent neuroanatomical evidence, which indicates a high degree of intrinsic excitatory connectivity in the PF cortex (Pucak et al., 1996). It has also been suggested that neurons with similar response preferences in the PF cortex are interconnected (Pucak et al., 1996). This suggestion is not inconsistent with our results that the same kind of selective units had strong recursive connections.

Of course, we do not claim that the model network mimics the biological one precisely, or that the learning algorithm replicates the learning mechanism of the neuronal system. However, we can say that the mechanisms for computing the input-output functions may be similar to various types of distributed information-processing networks centering around the PF cortex.

In summary, the present simulation results demonstrate that rule-selective neurons play a functional role in the coding and retention of rule information, and this information is used to produce the correct response appropriate to the context.

References

- [1] W. F. Asaad, G. Rainer, and E. K. Miller. Task-specific neural activity in the primate prefrontal cortex. *Journal of Neurophysiology*, 84:451–459, 2000.
- [2] S. Funahashi and M. Inoue. Neuronal interactions related to working memory processes in the primate prefrontal cortex revealed by cross-correlation analysis. *Cerebral Cortex*, 6:535–51, 2000.
- [3] J. M. Fuster, M. Bodner, and J. K. Kroger. Cross-modal and cross-temporal association in neurons of frontal cortex. *Nature*, 405:347–351, 2000.
- [4] E. Hoshi, K. Shima, and J. Tanji. Task-dependent selectivity of movement-related neuronal activity in the primate prefrontal. *Journal of Neurophysiology*, 80:3392–3397, 1998.
- [5] T. Minami and T. Inui. A neural network model of working memory: Processing of “what” and “where” information. In J. Mira and A. Prieto eds., *IWANN*, Vol. 2084 of *Lecture Notes in Computer Science*, pp. 126–133. Springer, 2001.
- [6] J. Mitchell and D. Zipser. A model of visual-spatial memory across saccades. *Vision Research*, 41, 2001.
- [7] S. L. Moody and S. P. Wise. A model that accounts for activity prior to sensory inputs and responses during matching-to-sample tasks. *Journal of Cognitive Neuroscience*, 12(3):429–448, 2000.
- [8] S. L. Moody, S. P. Wise, G. di Pellegrino, and D. Zipser. A model that accounts for activity in primate frontal cortex during a delayed matching-to-sample task. *Journal of Neuroscience*, 18, 1998.
- [9] R. Passingham. *The Frontal Lobes and Voluntary Action*. Oxford University Press, 1993.
- [10] M. L. Pucak, J. S. Levitt, J. B. and Lund, and D. A. Lewis. Patterns of intrinsic and associational circuitry in monkey prefrontal cortex. *Journal of Comparative Neurology*, 376:614–30, 1996.
- [11] G. Rainer, W. F. Asaad, and E. K. Miller. Memory fields of neurons in the primate prefrontal cortex. *Proc. Natl. Acad. Sci. USA*, 95:15008–15013, 1998.
- [12] G. Rainer, W. F. Asaad, and E. K. Miller. Selective representation of relevant information by neurons in the primate prefrontal cortex. *Nature*, 393:577–579, 1998.
- [13] S. C. Rao, G. Rainer, and E. K. Miller. Integration of what and where in the primate prefrontal cortex. *Science*, 276:821–824, 1997.
- [14] J. D. Wallis, K. C. Anderson, and E. K. Miller. Single neurons in the prefrontal cortex encode abstract rules. *Nature*, 411:953–956, 2001.
- [15] I. M. White and S. P. Wise. Rule-dependent neuronal activity in the prefrontal cortex. *Experimental brain research*, 126:315–35, 1999.
- [16] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280, 1989.
- [17] J. Xing and R. A. Andersen. Memory activity of lip neurons for sequential eye movements simulated with neural networks. *Journal of Neurophysiology*, 12(3):429–448, 2000.
- [18] D. Zipser. Recurrent network model of the neural mechanism of short-term active memory. *Neural Computation*, 3:179–193, 1991.
- [19] D. Zipser. Identification models in the nervous system. *Neuroscience*, 47:853–862, 1992.
- [20] D. Zipser and R. A. Andersen. A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. *Nature*, 331:679–684, 1988.

Selection of Models for Time Series Prediction via Meta-Learning

Ricardo Bastos Cavalcante Prudêncio, Teresa Bernarda Ludermit
Centro de Informática, Universidade Federal de Pernambuco
Caixa Postal 7851 - CEP 50732-970 - Recife (PE) - Brasil
[rbcp, tbl]@cin.ufpe.br

Abstract. In this work, we propose the use of meta-learning techniques in the task of selecting models for time series prediction. In our approach, a machine learning algorithm generates symbolic knowledge used for choosing a better model to predict a time series, according to the features of the series. In the implemented prototype, a decision tree is used for selecting between the Simple Exponential Smoothing model and the Time-Delay Neural Network, for predicting stationary time series. Our experiments revealed encouraging results.

1. Introduction

Time series prediction has been used in several real world problems, in order to eliminate losses resultant from uncertainty, as well as to support the decision-making process [1]. Several time series prediction models can be used to predict a time series. Selecting the most adequate model for a given time series, from a set of available models may be a difficult task, depending on the candidate models and the time series characteristics.

A force-brute approach consists of trying out each model using the available data, and selecting the one which obtained the best results. Although this approach is very straightforward, it is very costly when there is a large number of candidate models, and in the case of complex models.

A knowledge-oriented solution for the task of model selection can be found in [2], where the authors developed a base of rules that combined the forecasts of four simple time series models, according to features of the time series. These rules were developed based on the knowledge provided by five experts in the area. Despite the good results of this solution, developing rules for model selection may become an unfeasible task, since good experts are not always available and it is very difficult to acquire knowledge about complex models.

In our work, we propose the use of a machine learning algorithm to acquire symbolic knowledge for selecting time series prediction models. The solution proposed in this work uses ideas of the Meta-learning approach [3], which aims to select the best available algorithm for a machine learning problem based on the characteristics of the problem. A Meta-learning technique uses empirical examples to generate a machine learning model (*meta-learner*) which relates the performance of a set of candidate algorithms (*base-learners*) to the characteristics of the problem. Each training example consists of a machine learning problem, represented by a set of characteristics, associated to the performance empirically obtained by the *base-learners*. In this work, we considered the time series

prediction as a machine learning problem and adapted the meta-learning ideas to the problem of selecting time series models.

In our experiments, we used a decision tree induction algorithm as the meta-learner to select between two time series models: the Simple Exponential Smoothing [1] and the Time-Delay Neural Network [4]. In these experiments, the decision trees were induced using 99 time series problems extracted from literature. Our results revealed a significant improvement in the selection task (around 10%) compared to a naive classifier (corresponding to a situation where no knowledge is available to select the models).

In section 2, we present some approaches to time series model selection. Section 3 brings a brief explanation about Meta-Learning and some of its techniques. In section 4, we present our approach to select time series models using meta-learners, followed by the description of the implemented prototype. Section 5 presents the experiments and the results obtained by this prototype. Finally, we have the conclusion and future work in section 6.

2. Time Series Model Selection

According to [5], a time series is a set of observations generated sequentially in time. We can define the time series prediction problem as the estimation of the future values of the time series using the past values of that series. Time series forecasting can be used to support the decision-making in several application domains, such as finance, industry, among others [1]. There are several techniques that can be used to predict time series, such as the family of Exponential Smoothing models [1], the Box-Jenkins models [5] and several Artificial Neural Networks models [6]. Despite the diversity of existing models, there is no single technique that performs better than the others in all cases [2]. The adequacy of a model depends on the features of the time series to be predicted and on the characteristics of the application, such as constraints of time and resources.

The force-brute solution to the model selection task is simply to try out all the candidate models to the problem at hand, comparing the performance obtained on the available data. However, this solution is costly and not feasible for applications with a large amount of data or with several candidate models. The use of (theoretical or empirical) knowledge could drastically reduce the effort involved in the task of algorithm selection.

An approach that uses knowledge in model selection is based on the development of expert systems, such as the Rule-based forecasting system [2]. In that work, the authors implemented an expert system with 99 rules used to configure and combine the following time series models: Random walk, Linear regression, Holt's linear exponential smoothing and Brown's linear exponential smoothing. The rule base was developed based on the guidelines provided by five human experts through the analysis of real problems. The authors used time series features in the rules (such as level discontinuities, insignificant basic trend, last observation unusual, among others) to modify the weight associated to each model. Figure 1 presents a rule implemented in that work, considering the presence of level discontinuities in the series. In this rule-based approach, the authors focused on very simple models. For more complex models, it is difficult to find experts capable of giving useful insights.

<p>IF there are level discontinuities THEN add 10% to the weight on the random walk and subtract it from that on Brown's and Holt's.</p>
--

Figure 1: Example of a rule implemented in the Rule-Based Forecasting System

An approach to acquire empirical knowledge is to perform competitions among models on different sets of time series, and to analyze the obtained results. This approach was used, for example, in the famous M-Competition [7]. The drawback here is that the analysis is performed by human experts, who may not be able to identify some implicit knowledge embedded in the competition results. A human expert may not be efficient in analyzing innumerable hypothesis that can be tried out to explain the results, and may not be well-succeeded in generalizing situations.

Furthermore, the insights obtained from these competitions, in general, have not been defined in precise terms, as the rules developed in the Rule-based forecasting system. In the M-Competition, for example, the authors suggested that simple methods performed just as well as more sophisticated procedures. However, this statement is too general and imprecise, and it does not say when a user should choose a more complex model instead of a simpler one. As it will be seen, our proposal represents an step forward in relation to the competition approach, since we analyze the empirical examples deploying a machine learning algorithm which returns symbolic knowledge expressed in precise terms, such as rules or a decision tree.

3. The Meta-Learning Approach

There are several machine learning algorithms that can be applied to a learning problem. As there is no single best algorithm for all possible cases [8], an important question is how to choose the best algorithm to a given problem. Meta-learning emerges in this scenario as a promising approach to automatically acquire empirical knowledge that can support non-expert users in the algorithm selection task.

According to [3], there are different interpretations of the term “Meta-Learning”. In our work, we understand Meta-Learning as the process of generating knowledge that relates the performance of machine learning algorithms to the characteristics of the problem (i.e., characteristics of its data sets). This process is performed using empirical examples. Each example consists of a machine learning problem represented by a pre-defined set of characteristics (*meta-features*) associated to the performance previously obtained by the algorithms (*base-learners*). A set of such examples serves as a basis to the application of another machine learning algorithm (*the meta-learner*), responsible for discovering knowledge which will be useful for selecting the most adequate algorithms to a problem.

One of the first works that clearly stated the ideas of Meta-Learning was [9] in the context of the STATLOG project. In this work, the authors applied 20 machine learning algorithms to 22 benchmark classification problems. For each algorithm, they generated one meta-learning problem consisting of a set of 22 training examples, each one associated either to the class ‘applicable’ or to the class ‘inapplicable’. The class ‘applicable’ was assigned when the classification error obtained by the algorithm fell within a pre-defined confidence interval, whereas ‘inapplicable’ was assigned otherwise. Each problem was described by a set of 16 meta-features, from simple and statistical to information theory measures of the data sets. Finally, they applied in the meta-level the C4.5 algorithm to induce a decision tree. Although the authors used a very small set of problems to induce the meta-learners, this work was a landmark in the meta-learning field.

Another important work is the NOEMON approach [10], where each pair of base learners generated a particular meta-learning problem. Given the pair (X, Y) of base learners, a problem was classified within one of the classes ‘X’, ‘Y’ or ‘equal’ according to the performance obtained by the algorithms on that problem. The difference of performance was measured using a statistical test. Given a new problem, the NOEMON system could

provide a ranking of algorithms by combining the classifications given by the meta-learners. Furthermore, they provided a mechanism which enabled the definition of a different set of meta-features for each pair of algorithms. Due to these characteristics, this system is more flexible than the STATLOG approach. The NOEMON's prototype used the k-NN algorithm as the meta-learner.

The Zoomed-Ranking approach [11] provides a ranking of algorithms using the k-NN algorithm. It assumes that the classification or the ranking of algorithms when applied to a specific problem will be similar to the ranking of those algorithms when applied to a similar problem. An important feature of this work was that it used a multi-criterion measure to evaluate the base-learners, with the accuracy and execution time of the algorithms as parameters of a global performance measure.

An interesting approach to define meta-features is to use the performance obtained by very simple algorithms, called *landmarkers* [12]. The idea here is to relate the performance of the base-learners to more simple and faster designed algorithms. This approach claims that some widely used meta-features are very time consuming in some cases, even more than the base-learners. Hence, landmarking would be an economic approach to the data characterization step and could provide useful information for the meta-learning process.

4. Meta-learning for Model Selection

As seen in section 2, one way to select models for time series prediction is using expert systems. The knowledge used in this kind of system is extracted from human experts and it is expressed in the form of rules. Unfortunately, the process of knowledge acquisition depends on the availability and experience of experts. In our work, we propose to use a machine learning algorithm to automatically acquire knowledge for the selection of time series models. By treating time series prediction as a machine learning problem, our proposal falls within the meta-learning approach commonly used to select algorithms for classification problems. Here, the model selection can be seen as the task of choosing the learning algorithm, and the time series models corresponds to the base-learners.

In order to facilitate the knowledge acquisition process, we propose the use of specialized systems to select models for predicting the following time series types: (1) stationary time series, which has no trend or seasonal behavior, (2) series presenting trend, (3) seasonal series, and (4) seasonal series with trend. These types of series have commonly been analyzed in separate by the time series analysis community (as in [5]). In fact, the set of characteristics describing time series, as well as the models that are plausible for each type, are different.

In our proposal, each type of time series is associated to a set of *candidate models*. Given a time series to be predicted, the first step is to identify its type. This identification step can be performed using statistical tests to verify the presence of trend or seasonality in the series (such as the Wald-Wolfowitz test for trend and the Kruskal-Wallis test for seasonality [13]). Following, the corresponding specialized system suggests to the user one of the previously associated candidate models, or a ranking of those models.

This proposal was originated from a previous work in which we used a base of examples to select the architecture of neural networks for time series prediction [14]. Although that work already deployed some implicit ideas of meta-learning, such as the use of features to associate time series to neural architectures, it was not clearly associated to the meta-learning approach. The work presented here can be seen as an extension of this previous work, in which we use the ideas applied in neural network design to tackle the problem of time series model selection.

4.1 The implemented prototype

In order to verify the viability of our proposed approach, we implemented and tested a prototype. The experiments' results are shown in section 5. Due to time constraints, the initial prototype focused only on stationary time series. The choice of this type of series is due the fact that even non-stationary time series can be reduced to stationary series, as it can be seen in [5] and [6].

Considering that it is more useful to discriminate between prediction models with different behaviors than between models which have similar performances, our prototype implemented as candidates two models from different families: Simple Exponential Smoothing (SES) [1] and the Time-Delay Neural Network (TDNN) [4]. Both models are well indicated for predicting stationary time series, however they bear very distinct characteristics: while the former is a very simple and linear model, the latter is a more complex and non-linear model. As we will see in section 5, the quality of the prediction obtained with these two models may be very different, depending on the time series being predicted. Another reason for considering a neural network model is that, although it represents a powerful and promising approach, there is not much theoretical and empirical knowledge to guide its usage.

In figure 2, we can see the architecture of the implemented prototype. As said, we only implemented the specialized system for stationary time series. As it is usual in machine learning systems, this prototype has two phases: *training* and *use*. In the phase of *use*, given a new time series to predict, the *Feature Extractor (FE)* extracts the values of the time series features. According to these values, the *Knowledge-Base Processor (KBP)* module suggests to the user either the SES model or the TDNN model. For that, the KBP module maintains a set of rules previously provided by the *Meta-Learner (ML)* module as a result of the *training* phase. These rules are induced from a set of examples stored in a *Database (DB)* by the machine learning algorithm implemented in the ML module. These rules may be refined as more examples are available in the DB.

4.1.1 The Feature Extractor

A time series can be characterized by measures commonly used in time series analysis, such as length, autocorrelations, coefficient of variation, among others. In our research, we observed that the set of meta-features used to describe a type of time series may be different from those describing another type. For example, whereas the feature *period of seasonality* may be useful for describing seasonal series, it is useless for stationary series.

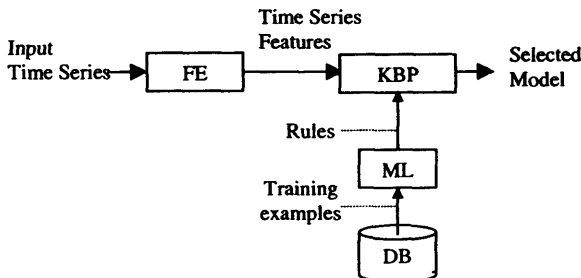


Figure 2: Architecture of a specialized system.

Table 1: Meta-features for time series prediction

Feature	Description	Type
len	Length of the time series	Integer
coef_var	Coefficient of variation	Real
mean_cor	Mean of the 5 first serial autocorrelation	Real
test_cor	Presence of significant autocorrelations	{Yes, No}
test_cor1	Test of significance of the first autocorrelation	{Yes, No}
test_cor2	Test of significance of the second autocorrelation	{Yes, No}
test_cor3	Test of significance of the third autocorrelation	{Yes, No}
abs_skew	Absolute value of the skewness coefficient	Real
abs_kur	Absolute value of the kurtosis coefficient	Real
turn_points	Test of turning points for noise in the time series	{Yes, No}

In the context of the classification problem, tools defining standard meta-features have already been developed and used for meta-learning in different experiments. This is the case of the *Data Characterization Tool*, developed within the METAL project [15]. Unfortunately, for the time series prediction problem, there is no such standard set of meta-features, since the application of meta-learning to this problem is recent. Hence, we had to define our own set of meta-features extracted from the literature of time series analysis. In our prototype, we initially used a set of 10 descriptive meta-features (see table 1). As this set is possibly not optimal, in future implementations we will consider new features.

4.1.2 The Database

The Database stores training examples of time series prediction problems. Each example has two parts: (1) the *meta-features* describing the time series, which are those presented in table 1; and (2) the *class* attribute, which has one of the values 'ses' or 'tdnn'. In order to assign the class attribute, we applied both the SES and TDNN models on that series, and used its last 30 data points to test the models. We compared the Mean Absolute Error (MAE) obtained by each model on these 30 points, and assigned to the class attribute the model which obtained lower MAE.

4.1.3 The Meta-Learner

The Meta-Learner module implements a machine learning algorithm that will generate rules for the KPB module. It receives as input the training examples stored in the DB, and returns a set of rules associating characteristics of these time series to the adequate model. In our prototype, we used the J48 algorithm implemented in the WEKA Java package [16], which is a variant of the C4.5 algorithm proposed in [17]. Rules can be extracted from the induced tree simply by transcribing each branch of the tree as a rule.

Although a great number of works in the meta-learning field implements the k-NN algorithm, we opted to use an algorithm to learn rules. The main advantage of rules is that the learned knowledge is expressed in a symbolic language, which is not the case of k-NN. This characteristic facilitates the interpretation and validation of the acquired knowledge.

4.1.4 The Knowledge-Base Processor

The aim of this module is to match the features of the input time series with the conditions in the antecedent of the rules. After the matching, the module returns to the user the class indicated by the selected rule.

5. Experiments and results

We describe here the experiments that evaluated the performance of our prototype. We collected 99 time series available for download from the Time Series Library [18]. Most of them were originally presented in books of time series analysis, and are used as benchmark problems in the forecasting field. A data set of 99 training examples is considered large enough to perform reliable experiments (as seen in [19], chapter 5).

Some of the collected time series were originally non-stationary. In these cases, we applied difference operators to the series, as suggested by [5], in order to generate the corresponding stationary series. In the case of time series presenting trend, we applied the simple difference operator (see equation 1), and for seasonal series, we applied the seasonal difference operator (see equation 2). After the application of these operators, the resulting training set consisted of 99 stationary time series.

(Equation 1) $Z^d(t) = Z(t) - Z(t-1)$, where Z is the original series and Z^d is the transformed series.

(Equation 2) $Z^s(t) = Z(t) - Z(t-p)$, where Z is the original series, p is the period of seasonality and Z^s is the transformed series.

In table 2, we present the number of examples associated to each class, after testing each series with the two considered models ('ses' and 'tdnn'). We also show the classification error obtained by the naive classifier (*default error*), which always associates to a new example the class with more training examples (in our case, the class 'tdnn'). This kind of classifier is only useful when there is no information available to guide our choice for one of the models.

In the section 5.1, we evaluate the quality of the classifications performed by the meta-learner, and in section 5.2, we interpret the knowledge acquired by the meta-learner.

Table 2: Class distribution for our problem

# Instances classified as 'tdnn'	52
# Instances classified as 'ses'	47
Default error	47.47%

Table 3: Results of the leave-one-out procedure applied to J.48 algorithm

# of Correctly Classified Instances	62
# of Incorrectly Classified Instances	37
Total Number of Instances	99
Average Error	37.37 %
Standard Deviation of Errors	4.86 %
95% Confidence Interval of the Errors	[27.84%, 46.90%]

5.1 Classification performance

We evaluated the classification performance of the meta-learner using the *leave-one-out* approach [19]. At each step, 98 examples are used as the training set, and the remaining example is used to test the generated decision tree. This procedure is repeated 99 times, using at each time a different test example. The algorithm is then evaluated based on the obtained test errors.

In this experiment, we observed a number of 62 correctly classified examples from the set of 99 test examples, which means an average error of 37.37% (see table 3). Although the achieved error may be considered high in absolute terms, it is significantly lower than the default error of 47.47%, representing a gain of 10.10% in the classification task. This improvement can also be statistically confirmed by analyzing the 95% confidence interval of the test errors obtained by the meta-learner. This interval has 46.90% as the upper bound, which is lower than 47.47%.

5.2 Inspection of the rules

After analyzing the meta-learner's performance, we executed the J.48 algorithm using all 99 examples, and examined the induced tree, in order to interpret the acquire knowledge. Initially, we tried to identify the more relevant features for classifying the SES and TDNN models. This was done by examining the nodes in the first levels of the tree, which contain more discriminating attributes [19]. Up to the third level, the nodes had employed conditions concerning the variables 'mean_cor', 'test_cor1', 'test_cor2', 'test_cor3', and 'coef_var'. The most discriminating feature was the 'mean_cor', which was selected in the root of the tree.

The generated tree presented a number of 16 leaves, which corresponded to 16 rules. Some of these rules can be better interpreted, such as the Rule 1 (see figure 3). The conditions 'mean_cor < 0.18' and 'test_cor3 = 0' may indicate a low degree of correlation on the series, and the condition 'coef_var <= 5.15' can be interpreted as indicating a low degree of uncertainty. When a series presents the conjunction of these conditions, the rule suggests that TDNN would be more adequate than the SES model.

Unfortunately, most of the rules cannot be easily interpreted. This is the case of the rules with more than 3 conditions. In future work, we will investigate the use of pruning techniques in order to generate simpler trees without affecting the classification performance.

<p>Rule 1: IF mean_cor < 0.18 AND test_cor3 = 0 AND coef_var <= 5.15 THEN Class = 'tdnn'</p>
--

Figure 3: Rule extracted from the decision tree.

6. Conclusion

In this work, we proposed a new approach to provide knowledge for the selection of time series prediction models. We can point out contributions of this work to two different fields: (1) in the *meta-learning field*, we applied meta-learning concepts to a problem which had not yet been tackled: the time series prediction. Although the concepts of meta-learning may be applied to different machine learning tasks (for example, clustering and regression problems), the previous efforts were mainly focused on the classification task; and (2) in the *time series prediction field*, we provided a new approach to acquiring empirical knowledge that can be used to optimize the prediction performance and to provide a better understanding of the time series models' behavior.

In our implemented prototype, we used decision trees to choose between Simple Exponential Smoothing and Time-Delay Neural Network models for predicting stationary time series. In the experiments, the precision of the selection task was significantly improved when the meta-learner was used.

As future work, we will implement other meta-learners for different types of time series, not only for stationary ones. Furthermore, we may perform some modifications in the current implementation, such as to augment the set of meta-features describing the time series, to augment the set of candidate models, and to modify the current meta-learner in order to obtain simpler rules.

References

- [1] D. C. Montgomery, L. A. Johnson and J. S. Gardiner, *Forecasting & Time Series Analysis*, Mc-Graw-Hill, New York, NY, 1990.
- [2] F. Collopy and J.S. Armstrong, Rule-based Forecasting: Development and Validation of an Expert Systems Approach to Combining Time Series Extrapolations, *Management Science*, Vol. 38, no. 10, pp. 1394-1414, 1992.
- [3] R. Vilalta and Y. Drissi, Research Directions in Meta-Learning, *Proceedings of the International Conference on Artificial Intelligence (ICAI01)*, Las Vegas, Nevada, 2001.
- [4] K. Lang and G. Hinton, Time-Delay Neural Network Architecture for Speech Recognition, *CMU Technical Report CS-88-152*, Carnegie-Mellon University, Pittsburgh, PA, 1988.
- [5] G. E. Box & G. M. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden-Day, San Francisco, CA, 1970.
- [6] G. Dorffner, Neural Networks for Time Series Processing, *Neural Network World*, Vol. 6, No. 4, pp. 447-468, 1996.
- [7] S. Makridakis, A. Andersen, R. Carbone, R. Fildes, M. Hibon, R. Lewandowski, J. Newton, E. Parzen and R. Winkler, The Accuracy of Extrapolation Time Series Methods: Results of a Forecasting Competition, *Journal of Forecasting*, Vol. 1, No. 2, pp. 111-153, 1982.
- [8] P. Brazdil and C. Soares, A Comparison of Ranking Methods for Classification Algorithm Selection, *Proceedings of the 11th European Conference on Machine Learning (ECML2000)*, pp. 63-74, Springer, 2000.
- [9] D. Michie, D. J. Spiegelhalter and C. C. Taylor, *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, New York, 1994.
- [10] A. Kalousis, G. Zarkadakis, and T. Theoharis, Noemon: Adding Intelligence to the Knowledge Discovery Process, *Proceedings of the 17th SGES International Conference on Knowledge Based Systems and Applied AI*, pp. 235-249, SGES Publications, 1997.
- [11] C. Soares and P. Brazdil, Zoomed Ranking: Selection of Classification Algorithms Based on Relevant Performance Information, *Proceedings of the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD2000)*, pp. 126-135, Springer, 2000.
- [12] B. Pfahringer, H. Bensusan, and C. Giraud-Carrier, Meta-learning by Landmarking Various Learning algorithms, *Proceedings of the 17th International Conference on Machine Learning (ICML-2000)*, pp. 743-750, Stanford, CA, 2000.
- [13] P. A. Morettin and C. M. Toloi, *Séries Temporais, Métodos Quantitativos*, Atual, São Paulo, SP, 1987.

- [14] R. B. C. Prudencio and T. B. Ludermit, Design of Neural Networks for Time Series Prediction Using Case-Initialized Genetic Algorithms, *Proceedings of the 8th International Conference on Neural Information Processing* (ICONIP'01), pp. 990-995, Shanghai, China, 2001.
- [15] ESPRIT METAL Project (26.357) [<http://www.cs.bris.ac.uk/~cgc/METAL>].
- [16] I. H. Witten and E. Frank, *WEKA: Machine Learning Algorithms in Java*, University of Waikato, Hamilton, New Zealand, [www.cs.waikato.ac.nz].
- [17] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA, 1993.
- [18] Time Series Library, [<http://www-personal.buseco.monash.edu.au/~hyndman/TSDL>].
- [19] T. Mitchel, *Machine Learning*, MacGraw Hill, New York, NY, 1997.

Spiking Neurons in Clustering of Diabetic Retinopathy Data

Krzysztof J. Cios^{1,2,3,4}, William Jackson⁵, Waldemar Swiercz² and Laura Springhetti⁵

¹University of Colorado at Denver

²University of Colorado at Boulder

³University of Colorado Health Sciences Center; Denver, CO

⁴4cData LLC, Golden, CO

⁵Barbara Davis Center for Juvenile Diabetes, Denver, CO

Abstract In simple models of biological neurons the output does not depend on time. In contrast, the spiking neuron model, in response to external stimulation, generates a series of action potentials (spikes). In the paper we use MacGregor's spiking neuron model, and the Temporal Correlation Learning (TCL) rule to update synaptic connections between spiking neurons. The network of such neurons, with the TCL rule, was shown to be capable of clustering, without the need of specifying the number of clusters. In this paper the network of spiking neurons is used for finding clusters in eye images of diabetic retinopathy patients. Diabetic retinopathy is a disease caused by diabetes that if not treated can lead to major loss of vision.

1. Introduction

Diabetic Retinopathy (DR) is caused by diabetes and is one of the major causes of blindness. Elevated blood sugar in diabetics causes the blood vessels in the retina to leak, close, and proliferate, damaging the retina. The vessels can rupture and release blood into the vitreous thereby causing cloudy vision. Nearly all patients with diabetes will eventually develop DR[1], but they will not know that they have it until the later stages of the disease. Even though there is higher risk of severe DR in type 1 diabetes, there are more type 2 diabetes cases, thus more patients with visual loss in this group. Type 1 diabetes is usually juvenile onset insulin dependent, and type 2 is usually adult onset non-insulin dependent. If not properly treated, DR can cause blindness, but with appropriate treatment over 90% of visual loss can be prevented. Prevention of DR is better not only for patient's health but it can also save money. Thus, diagnosis of early stages of DR is very important. To achieve this very goal every diabetic patient should be examined yearly. The reason is that the patient will not develop severe DR before being detected by the next examination, and appropriately treated. Unfortunately, about 50% of diabetic patients were never examined and only 30% of them were examined regularly[2].

Automatic computer recognition of DR through image analysis may provide for early detection. More patients can be examined since only a technician is required to do the screening. Patients do not have to travel to a, possibly distant, ophthalmology center so they can be examined more often. Only patients diagnosed positively for sight threatening DR will be required to see the ophthalmologist.

Digital images of the fundus are used to detect microaneurysms, hemorrhages, exudates, and neovascularization that indicate positive diagnosis of DR. Depending on the pattern of abnormalities, the severity level of the disease is diagnosed. In contrast to the "gold standard" of seven Early Treatment Diabetic Retinopathy Study (ETDRS) field

classification, the method presented here uses three fields per eye, namely nasal, optic nerve and macula, and the temporal field, to recognize and grade the extent of DR.

2. Images and their Preprocessing

Images of fundus of the eye are images of 2160x 1140 pixels in a 24 bit color. Since the green layer of the image contains the most information (red and blue layers contain almost no information) only the green layer is used. The color, contrast and brightness of each image are distinct for each patient, as well as non-uniformity of the image intensity. To diminish the influence of those factors, two filters are used:

- Center Weighted Median (CWM) filter to remove noise[3].
- Flat Fielding (FF) filter based on fast Fourier transform (FFT) to make intensity of the images uniform[4].

The CWM filter is very effective in removing noise from images. Its advantage over the simple Median Filter (MF) is that it prevents image details and allows for good noise removal. The CWM works almost like the MF. The difference is that all pixels, except for the center one, are assigned a weight value of 1. The central pixel of the processed window is assigned a weight bigger than one. The value of that weight specifies the strength of filtering. After sorting all the pixels in the window, the selection of median value takes place. Beginning with the pixel with the highest value the corresponding weights are added until the sum of the weights is bigger than or equal to the average of the all weights. The value of the pixel for which the addition stopped is returned as a median value.

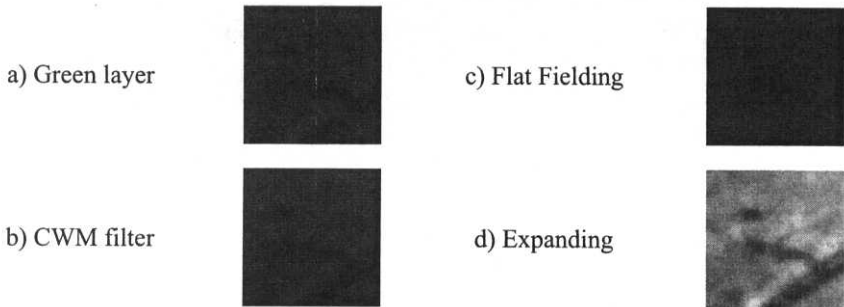


Figure 2.0 Image preprocessing steps.

FF filter helps in reducing non-uniform intensity of the image. First, the original image is filtered using low-pass filter based on the FFT. Then the low-pass filtered image is subtracted from the original image. This operation leaves only high and medium frequency features on the image.

After applying the filters to the image, its histogram was expanded to increase the contrast. Preprocessed images are then used as input for clustering using a network of spiking neurons.

However, large dimension of the images makes it almost impossible to perform computations on the entire image. To address this problem, we decided to process only samples of the images of the size 32x32 pixels. Although bigger dimension of the subimages can also be processed the time needed for computations increases nonlinearly.

Figure 2.0 illustrates five steps of image preprocessing on the sample image containing part of a blood vessel and a small hemorrhage.

3. Implementation of the Network of Spiking Neurons

As said above we will use the network of spiking neurons to recognize DR patient data. To define such a network we need to define the spiking neuron model, the plasticity rule, and the network topology. Since spiking neuron model and the plasticity rule are complex, we describe them in some detail below.

3.1 The Neuron Model

Biological neurons generate series of action potentials (spikes) in response to stimulation. Taking into account the spiking nature of biological neurons is crucial for their accurate modeling. The duration and frequency of spike generation depends on the strength of stimulus[5]. The most accurate artificial neuron model is Hodgkin-Huxley model that takes into account several ion channels. Additional accuracy is gained by considering compartmental geometry of a neuron (nucleus, dendrites, and axon)[6]. The side effect of higher accuracy of the neuron model is that it entails higher computational complexity, which can make designing networks using this type of a neuron model very inefficient. Because of that we decided to use MacGregor's model. It is less complex than Hodgkin-Huxley but is still models well the biological neuron behavior. It accounts for the neuron membrane potential, the potassium channel, and the excitatory and inhibitory synapses. MacGregor's[7] neuron model is described by equations 3.1.0 through 3.1.3:

Spike generation :

$$S = \begin{cases} 1 & E \geq T_h \\ 0 & E < T_h \end{cases} \quad (3.1.0)$$

Transmembrane potential :

$$\frac{dE}{dt} = \frac{-E + G_K \cdot (E_K - E) + G_e \cdot (E_e - E) + G_i \cdot (E_i - E) + SCN}{T_{mem}} \quad (3.1.1)$$

Refractory properties :

$$\frac{dG_K}{dt} = \frac{-G_K + B \cdot S}{T_{GK}} \quad (3.1.2)$$

Accommodation of threshold :

$$\frac{dT_h}{dt} = \frac{-(T_h - T_{h0}) + c \cdot E}{T_{Th}} \quad (3.1.3)$$

where:

Transmembrane potentials:

$$E = V - V_r \quad E_K = V_K - V_r$$

$$E_i = V_i - V_r \quad E_e = V_e - V_r$$

V – membrane potential

V_r – membrane resting potential

V_K – potassium resting potential

V_i – inhibitory resting potential

V_e – excitatory resting potential

Transmembrane conductances:

$$G_K = g_K / G$$

$$G_i = g_{si} / G$$

$$G_e = g_{se} / G$$

G – membrane resting conductance

g_K – potassium resting conductance

g_{si} – inhibitory resting conductance

g_{se} – excitatory resting conductance

SC – current injected to cell

T_{GK} – decay of G_K time constant

T_h – threshold value

T_{h0} – resting value of threshold

T_{th} – decay of threshold time constant

T_{mem} – membrane time constant

$T_{mem} = C/G$

Current through a membrane

$$SCN = SC/G$$

C – membrane capacitance

c – determines rise of threshold $c \in [0,1]$

B – postfiring potassium increment

The neuron membrane potential is changed relative to the input signal. Neuron fires when the membrane potential is higher than the threshold value. If the stimulus is small the membrane potential never reaches the threshold and the neuron does not fire. If the stimulus is strong enough to cause the membrane potential to reach the threshold, neuron fires and a spike is generated. Immediately, after spike generation, the neuron cannot respond to any stimulation; this time interval is known as the absolute refractory period. The following time interval is known as a relative refractory period, when neuron can respond only to a very strong stimulation. Depending on the neuron properties, the threshold value can accommodate to the signal, requiring stronger stimulus for the next spike generation. Figure 3.1.0 shows an example of the model's response to the step current stimulation. Figures 3.1.1 through 3.1.3 show the model responses to stimulation by series of spikes with different frequencies.

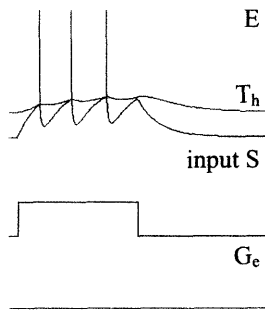


Figure 3.1.0 Model response to step current stimulation.

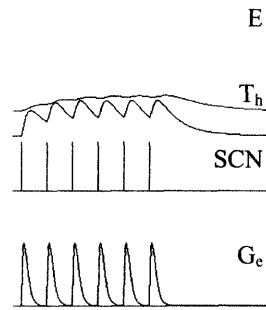


Figure 3.1.1 Model response to stimulation by series of spikes; 30ms space between spikes.

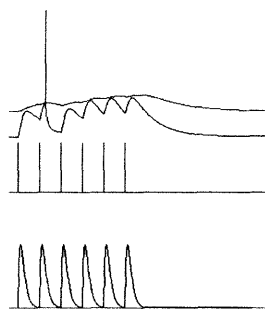


Figure 3.1.2 Model response to stimulation by series of spikes; 25ms space between spikes.

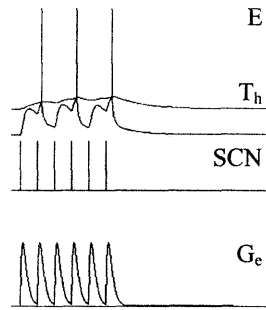


Figure 3.1.3 Model response to stimulation by series of spikes; 20ms space between spikes.

3.2 Network Topology

The topology of the network is defined as follows. Each pixel of the image is represented by a neuron in the network. Differences in intensities between neighboring pixels are represented as a time delay between respective neurons in the network. Thus, if the difference between two neighboring pixels is 50 then the time delay between the corresponding neurons is 50ms. If the pixels are not adjacent, the time delay is set to infinity. Two types of the neighborhoods are used, one with dimension 3x3 and the other with dimension 5x5, see Figure 3.2.0. The black circle represents the neuron under consideration, the gray circles represent adjacent neighboring neurons, and the white circles represent non-adjacent neurons.

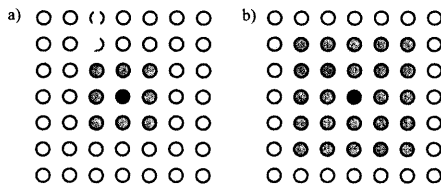


Figure 3.2.0 Neighborhood types: a) dimension 3x3, b) dimension 5x5.

Experiments are performed separately for each neighborhood type. For each experiment all neuron synapses (weights) are initially set to the same value, equal to about 50% of the maximum value. Value of the t_{corr} initially is set to a relatively big value, namely 70% of the maximum value. In each iteration the neurons are stimulated in a random manner. The weights of synapses between neurons are adjusted (rewarded or punished) according to the Temporal Correlation Learning (see below) plasticity rule. After each iteration the value of the t_{corr} is decreased, which results in decreasing the number of rewarded synapses in the following iteration, which causes many synapses to disappear. At the end of the calculation only part of the original synapses remains. Those remaining connections define clusters of neurons. The network of spiking neurons finds clusters automatically; no a priori specification of the number of clusters is required. Similar type of the network was used for solving graph problems and was proven capable of unsupervised clustering of the data[8]. Before we specify the TCL plasticity rule, we need to define the Post-Synaptic Potential (PSP). Spike arrival is causing the PSP generation in a postsynaptic neuron, which changes the membrane potential. The PSP between neurons i and j is described by equation 3.2.0:

$$g_j(t) = \frac{t - \Delta t_{ij}}{T_{ij}} \cdot \exp\left(-\frac{t - \Delta t_{ij}}{T_{ij}}\right) \cdot \exp(1) \quad (3.2.0)$$

where:

Δt_{ij} – propagation time,

T_{ij} – synapse time constant,

$\exp(1)$ – normalizing factor.

Depending on whether a synapse is excitatory or inhibitory, the potential is either positive (EPSP) or negative (IPSP), respectively. Figure 3.2.1 shows shapes of the EPSP and IPSP. We assume that the synapse is activated when g_{ij} reaches its peak. Thus the delay between presynaptic neuron spike generation and the synapse activation is $t_{ij} = \Delta t_{ij} + T_{ij}$.

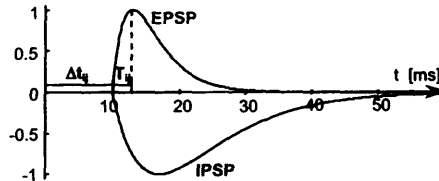


Figure 3.2.1 Postsynaptic potentials.

The total synaptic excitatory or inhibitory conductance of the neuron is obtained by calculating a weighted sum of all active PSP's[9].

3.3 Synaptic Plasticity Rule

We will use, within the network of spiking neurons, the Temporal Correlation Learning (TCL) rule of Sala and Cios[10] for modifying (learning) the strengths of the synapses (weights). The weight values are updated according to the TCL rule shown in equation 3.3.0.

$$w_{ij}(t+1) = \frac{w_{ij}(t) + \alpha \cdot c_{corr}(t_{ij})}{\sum (w_{ij}(t) + \alpha \cdot c_{corr}(t_{ij}))} \quad (3.3.0)$$

where:

w_{ij} – synaptic weight between firing of neurons i and j ,

α – learning rate,

c_{corr} – temporal correlation function.

Modification of the synapse weights, between pre- and post-synaptic neurons, takes place every time the postsynaptic neuron fires. The correlation function, used within the TCL rule, evaluates if a particular synapse is to be rewarded (synapse had influence on

postsynaptic neuron firing) or punished (synapse did not have influence on postsynaptic neuron firing).

The Temporal Correlation function is defined, after[7], by equation 3.3.1, and its shape is illustrated in Figure 3.3.0:

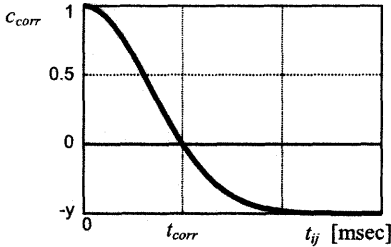


Figure 3.3.0 Temporal correlation function.

$$c_{corr} = (1 + y) \exp\left(-k \frac{t_{ij}^2}{t_{corr}^2}\right) - y \quad (3.3.1)$$

where:

t_{ij} – time elapsed between firing of j and i ;

t_{corr} – positive time window size;

y – max value for the decay, $y \in (0, 1]$;

$k = \ln(1 + 1/y)$ – assures that $c_{corr}(t_{corr}) = 0$

The following example illustrates how the TCL rule works. Let us consider two presynaptic neurons N_2 and N_3 and one postsynaptic neuron N_1 , see Figure 3.3.1. Value of t_{corr} is set to 5ms. Let us assume that postsynaptic neuron N_1 fires at time $t_1 = 0$. Presynaptic neuron N_2 fired 2ms before, and presynaptic neuron N_3 fired 7ms before. Since the time window size is 5ms, only neuron N_2 falls inside the window. Thus, according to the correlation function only the weight of the synapse (N_2, N_1) will increase, while the weight of synapse (N_3, N_1) will decrease. Figure 3.3.2 shows activation times and time window for the example.

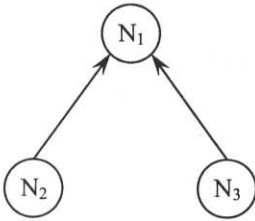


Figure 3.3.1 Postsynaptic neuron N_1 , and presynaptic neurons N_2 and N_3 .

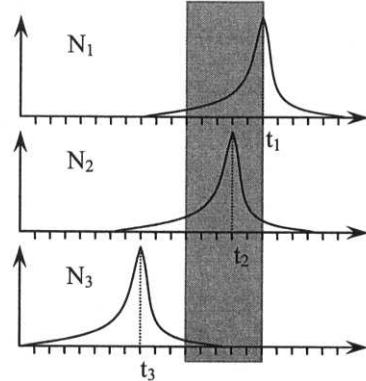
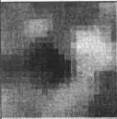
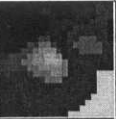
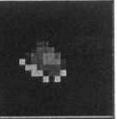

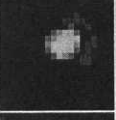
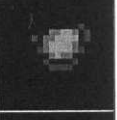

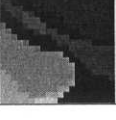
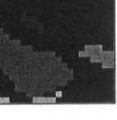
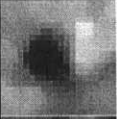
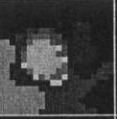
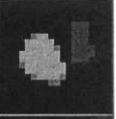
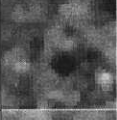




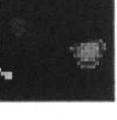


Figure 3.3.2 Activation times and time window

4 Results


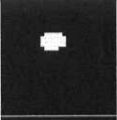


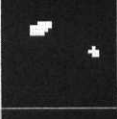
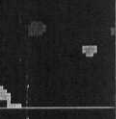

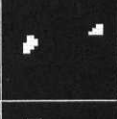


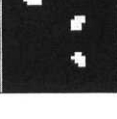

The results are presented as images since the goal is to distinguish between important objects of interest and background. Table 4.0 shows some of the results in a graphical form. Column 'Original' contains preprocessed subimages of size 32x32 pixels. Images 1, 2 and 4, contain hemorrhages, Image 3 part of a blood vessel, while Images 5 and 6 contain microaneurysms. The columns 'Neighborhood 3x3' and 'Neighborhood 5x5' show the results after clustering is performed. The size of the neighborhood influences the sensitivity of the network. The neighborhood 5x5 seems to produce results with higher specificity but lower sensitivity. For instance in Image 3, the blood vessel was not detected by the algorithm when using the larger neighborhood. On the other hand neighborhood 3x3 is less specific in detecting artifacts, what can be seen in Image 5. The results are satisfactory since they distinguish well between the image background and the objects of interest. The resulting images can be used as input to, say, Image Recognition Neural Network (IRNN)[11] to improve performance of DR recognition[12].

Table 4.0 Results

	Original	Neighborhood	
		3x3	5x5
Image 1			
Image 2			
Image 3			
Image 4			
Image 5			
Image 6			

We compared our results, achieved using the network of spiking neurons, with the results of the algorithm that uses the watershed transform to detect abnormalities in DR images[13]. The watershed transform results are shown after masking (removal) of blood vessels, and after pixel value checking, which removes non-relevant objects that have been detected. Again, we selected subimages of the size 32x32 pixels and compared the results. The neighborhood used in all computations was 3x3. Table 4.1 compares the results of the two algorithms.

Table 4.1 Results comparison

	Original	Watershed	Spiking
Image 1			
Image 2			
Image 3			
Image 4			

As can be seen the results of the network of spiking neurons algorithm are comparable with those of the watershed algorithm. It needs to be stressed, however, that they were obtained without masking blood vessels and without pixel value checking. Unfortunately, in some cases very low image contrast may cause detection of artifacts by our algorithm; for example see Image 3 in Table 4.1. In this case, the images need to be processed by a different algorithm that is able to remove the artifacts.

5 Discussion

The network of spiking neurons does a good job of clustering features of diabetic retinopathy images. The results are achieved in a fully automatic way; the number of clusters returned depends only on the data and does not need to be specified.

The comparison with the watershed algorithm showed that the network of spiking neurons algorithm generates good results, but in some cases the resulting images may require removal of artifacts. After the artifacts are removed, the images can be used as input to, for instance, the IRNN algorithm¹², which was used for DR detection from raw images. However, raw images contained lots of irrelevant information that had negative influence on the results (IRNN could not achieve more than 80% correctness for DR detection). The advantage of the network of spiking neurons is that it removes irrelevant information from images, thus its output can be used as an input to other, for example, classification algorithms.

The presented work is an original application of the network of spiking neurons to image segmentation. To speed up the algorithm either a hardware implementation would be needed or decreasing the algorithm's complexity; we plan to do the latter.

References

- [1] Aiello L.P. et al (1998). Diabetic Retinopathy, *Diabetes Care* vol.21, no1.
- [2] Taylor H.R. (1997). Diabetic Retinopathy: A Public Health Challenge, *American Journal of Ophthalmology*, vol.123, no 4.
- [3] Bovik A. (2000). Image and Video Processing, *Academic Press*.
- [4] Michael S. et al (2000). Practical Algorithms for Image Analysis. *Cambridge University Press*.
- [5] Kandel E.R. et al (2000) Principles of Neural Science, *McGraw-Hill Companies*.
- [6] <http://neuron.duke.edu>
- [7] MacGregor R.J. (1993). Theoretical Mechanics of Biological Neural Networks, *Academic Press Inc*.
- [8] Sala, D.M. and Cios, K.J. (1999). Solving Graph Algorithms with Networks of Spiking Neurons, *IEEE Trans. on Neural Networks*, vol. 10, no 4.
- [9] Cios, K.J. and Sala, D.M. (2000). Advances in Applications of Spiking Neuron Networks, *SPIE AeroScience 14th International Symposium, Applications and Science of Computational Intelligence III*, Orlando FL. pp. 324-336
- [10] Cios, K.J. and Sala, D.M. (2001). Networks of Spiking Neurons in Data Mining, *Pattern Recognition: From Classical to Modern approaches*, Pal S.K. Pal A. World Scientific. pp. 329-346
- [11] Cios, K.J., and Shin, I., (1995). Image Recognition Neural Network: IRNN, *Neurocomputing*, vol. 7 no 2.
- [12] Jackson, W., Cios, K.J., Swiercz, W., and Springhetti, L., (2002) Computer Analysis of Diabetic Retinopathy, *A Journal of the American Diabetes Association*, Vol.51, sup.2
- [13] Luo, G., Opas, C., et al, (2001) "Abnormality Detection in Automated Mass Screening System of Diabetic Retinopathy" *Computer-based Medical System 2001, Proceedings. 14th IEEE Symposium*

This page intentionally left blank

Section 3

Fuzzy Sets

This page intentionally left blank

A Fuzzy Relatedness Measure for Determining Interestingness of Association Rules

B. SHEKAR* and Rajesh NATARAJAN

(shek, rn)@iimb.ernet.in

*Quantitative Methods and Information Systems Area
Indian Institute of Management Bangalore
Bangalore 560076, INDIA*

Abstract

In Knowledge Discovery in Databases (KDD)/ Data Mining literature, 'interestingness' measures are used to rank rules according to the 'interest' a particular rule is expected to evoke in a user. In this paper, we introduce an aspect of interestingness called 'item-relatedness' to determine interestingness of item-pairs occurring in association rules. In actuality, association rules that contain weakly-related item-pairs are the ones that are interesting.

We elucidate and quantify three different types of item-relatedness. Relationships corresponding to item-relatedness proposed by us are shown to be captured by paths in a 'fuzzy taxonomy' (an extension of the concept hierarchy tree). We then combine these measures of item-relatedness to arrive at a total-relatedness measure. This total relatedness measure appropriately combines each aspect of relatedness-relationships among items. We finally demonstrate the efficacy of this total measure on a sample taxonomy. We analyse the results and explain intuitive correspondences between numerical results and reality.

1. Introduction

Knowledge Discovery in Databases (KDD) or Data Mining is a discipline that aims at extracting novel, relevant, valid and significant knowledge, expressed in the form of patterns from large and very large databases. Association rules that bring out co-occurrence of items in a database of transactions constitute one such pattern [9,10]. Due to the automated nature of the search process in association rule discovery, a large number of association rules are extracted – numbers that exceed human ability for easy comprehension. The use of 'interestingness' measures to rank the association patterns is one approach adopted by researchers to tackle this problem. Interestingness measures may be 'objective' or 'subjective' [1]. Objective measures classify the interestingness of a pattern in terms of its structure and the underlying data used in the discovery process [9], while subjective measures, in addition to the above, also try to incorporate the views of the person inspecting the pattern [1].

Here, we consider association rules discovered from market-basket transactions. We try to capture one aspect of subjective interestingness called 'item relatedness' using a fuzzy taxonomy (concept hierarchy) of items. Concept hierarchies have been used

* Corresponding author

extensively in data mining studies to find associations between items at different levels of the concept hierarchy [10], to find negative associations [2] and in attribute-oriented induction [5]. In our study, we deal with association rules whose items occur as leaves in the taxonomy tree. We include domain knowledge about items, their categories and their relationships in the structure of the taxonomy. We then use structural aspects of the taxonomy to derive an intuitive measure of 'relatedness'. Graff et al. [6, 7] have used taxonomies to find interesting association rules. They consider cases wherein multiple taxonomies are available for attributes [6] and attributes are fuzzy [7]. An item set (rule) is considered interesting if the behaviour of the item set varies considerably from the behaviour of its parents. They estimate/compute the support of the item set from the support of its parents. If the 'actual' support of the item set (obtained from the database of transactions) is different from this estimated support, then the item set is deemed 'interesting'. Our notion of interestingness is very different from this as it is based on the structural aspects of the taxonomy. In addition, we also fuzzify the relationships between items and their parents rather than the items/attributes themselves. Chen et al. [3] have used fuzzy taxonomies to mine three types of fuzzy association rules viz. rules with taxonomic nodes, linguistic terms and hedges. However, they do not deal with the relationships between items as done here, but are more concerned with mining generalized fuzzy association rules across the higher level taxonomic nodes. More importantly, we use the fuzzy taxonomy to quantify the interestingness of crisp association rules that are inherently non-fuzzy. The study that comes closest to our current study is due to Hamilton and others [5] who used concept hierarchies to rank generalized relations on basis of their interestingness. In their study, concept hierarchies were constructed for individual attributes, but they did not consider relationships that exist between the attributes themselves. The intuition used there was that nodes that lie between the extremes of the leaf nodes and the root node are most likely to provide new, non-trivial and previously implicit information. Here, we do not deal with attribute values explicitly, but rather consider them implicitly while constructing the 'fuzzy taxonomy'.

'Fuzzy taxonomy' is described in the next section along with three notions of item-relatedness. Subsequently we present a mechanism for calculating 'item-relatedness' existing between two items and discuss its appropriateness. We then discuss the implications of the new item relatedness measure and motivate it intuitively with the help of an example. Finally, we conclude with a summary of results.

2. Fuzzy Taxonomy

In a traditional concept hierarchy [5,10] each concept/category node can be the child of at the most one parent node. This restriction does not allow us to express certain relationships that might exist between items and their higher-level concepts. It becomes very difficult to categorize an item that might belong to two higher-level concepts at the same level [8]. Further, one also has cases wherein an item can be used in many ways, and might substitute other items to various extents. For example, a spoon can take the role of a knife to a limited extent for cutting soft items like butter and some kinds of fruits. The tip of a knife, made blunt, might serve as a screwdriver to a limited extent. Such cases of an item taking on multiple functions arise because of the presence of secondary functionalities in addition to the 'primary' functionality and the overlap of attributes required for the functions. Such relationships cannot be sufficiently expressed in a traditional concept hierarchy. In order to express the relationships arising out of secondary functionalities, we introduce the notion of 'fuzzy taxonomy'. We use standard graph-theoretic [8] and standard

terms from fuzzy literature [4] throughout. Specific terms are defined as and when they are introduced.

A 'Fuzzy Taxonomy', [3] an extension of the traditional concept hierarchy, is a tree with items represented as leaf nodes and concepts represented as non-leaf nodes. A 'Parent' and 'Child' pair represent an 'is-a' relationship. A 'child' node need not be a full member of the category/concept represented by the 'parent' node. In other words, some attributes of a 'child' node may not be relevant as compared to those of its 'parent'. Another important point to note is that there are no direct connections between siblings. Siblings are connected only through their parents.

We define a membership function that takes on a value between 0 and 1, and which represents the extent to which a 'child' node belongs to its 'parent' category. This value is then carried over to other ancestors (emanating from its parent) of the 'child' node. When an 'ancestor' node has two or more membership grades from the various parents of a leaf-level item, then we choose the highest membership value. This is done so as to preserve the membership value of the leaf-level item in its closest ancestor. Formally, the membership function of a child node 'c' in its parent node 'p' is given by: $\mu_{(c,p)} = x : 0 \leq x \leq 1$. For an ancestor 'a', the membership grade is given by $\mu_{(c,a)} = \max \{ \mu_{(c,a)}(k) \}$ where $k = 1, 2, \dots, N$. $\mu_{(c,a)}(k)$ is the membership grade of the child node 'c' in its ancestor 'a' by the virtue of the path 'k'.

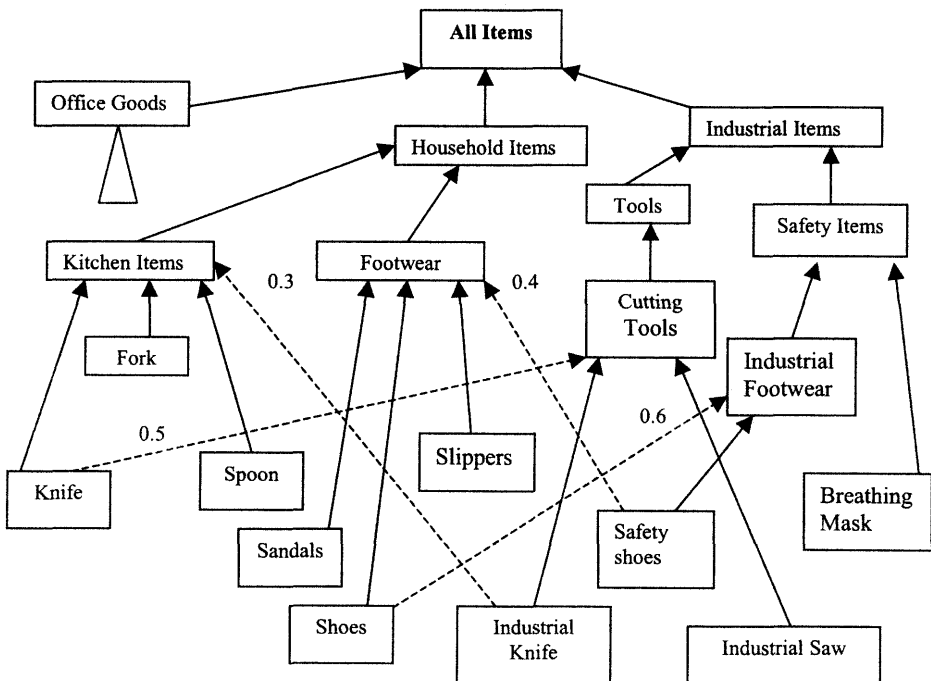


Figure 1: A Fuzzy taxonomy

Figure 1 shows a fuzzy taxonomy. Although it is somewhat compact to aid illustration, the discussions that follow can be extended to fuzzy taxonomies of any degree of complexity. Any item can have one or more parents. For example, node **knife** is a member of both the nodes **kitchen items** and **cutting tools**, although the membership of

knife in cutting tools is just 0.5, i.e. $\mu_{(\text{knife, cutting tools})}(k) = 0.5$ while $\mu_{(\text{knife, kitchen items})}(k) = 1.0$. We represent fuzzy membership grades (that is any membership less than 1.0) by broken lines with membership values written on them. Crisp memberships ($\mu=1.0$) are represented by solid lines. We note that the membership of **knife** in the node **industrial items** is 0.5 because of the membership being transferred to nodes **tools** and **cutting tools**. Thus, $\mu_{(\text{knife, Industrial items})} = 0.5$. On the other hand, the membership value of **knife** in the **All Items** node is 1, because of the direct crisp path through the node 'household items'.

$$\mu_{(\text{knife, All Items})} = \max \{ \mu_{(\text{knife, Household items})}, \mu_{(\text{knife, Industrial items})} \} = \max [1.0, 0.5] = 1.0$$
In a very broad sense the transfer gives the extent to which properties of the child overlap with those of the parent node. In order to get an idea of the relatedness between two items, we first need to find all simple paths [8] connecting the two items. A path here is a set of nodes connecting any two items.

Highest-level node of path [$H_{A, B}(p)$]: The highest-level node of path(A, B) is defined as the node that occurs at the highest level (i.e. nearest to the root node) in the path p connecting items A and B. We have a highest-level node for each of the paths connecting A and B. The highest-level node can also be defined in an alternative way. Consider the path from the item A to the root node say, path(A, Root), and the path from item B to the root node say, path(B, Root). The two paths will have at least one common node i.e. the 'root' node, but might have more common nodes. Then, the common node from the two paths viz. path(A, Root) and path(B, Root) which is closest to A and B is the highest level node of path(A,B). For example, in the crisp path between **knife** and **shoes**, the node **household items** is the highest-level node. By the virtue of its position, the highest-level node of a path plays a significant role in determining the 'relatedness' between the two items under consideration. It is the closest common context that relates the two items. Further, the distance of the highest-level node from the root node also gives an indication of the relatedness between the two items.

3. Item Relatedness

Relationships between items are a result of the generic category to which the items belong (primary functional purpose), domains of application and secondary functionalities. Any two items say A, B can be connected by many such relationships each of which increases the extent to which the two items are related to each other. It is also important to note that mere presence of a large number of relationships may not increase the relatedness between the two items since the strength of each of them also plays an important role in determining 'relatedness'. All association rules have support (statistical significance) and confidence (predictive ability) [9, 10] greater than user set minimum values. An item-pair in an association rule is very 'interesting' to a user if the items in it are weakly related to each other. The fact that these item-pairs, despite being weakly related, have a high frequency of occurrence in the database make them interesting. Thus, 'relatedness' and 'interestingness' are opposing notions. For example, an item pair {beer, diaper} will evoke interest because the items are weakly related, since they have generically different functionalities and are used in different domains of application. Here we discuss different views of item relatedness.

3.1 Highest-level Node Membership [$HM_{A, B}(p)$]

Consider the fuzzy path connecting items **knife** and **shoes**. The node **Industrial items** is the highest-level node of this path. Highest-level node membership of items A and

B in path 'p' is given by the minimum of the membership values of the two items in the Highest-level Node of path 'p'.

$$HM_{A,B}(p) = \min [\mu_{A,H(A,B)}(p), \mu_{B,H(A,B)}(p)]$$

Thus, $HM_{(knife, shoes)}(p) = \min [0.5, 0.6] = 0.5$

We consider 'minimum' as the operator that relates the memberships of the two items in the highest-level node. The membership value of **knife** in **Industrial Items** is 0.5 while that of **shoes** is 0.6. Therefore, both items have their presence in **Industrial Items** as members to the extent of at least 0.5. The rationale is that **knife** and **shoes** cannot be considered to be proper **Industrial items**, but they can be used together as 'industrial items' to a maximum extent of 0.5.

3.2 Highest-level Relatedness [$HR_{A,B}(p)$]

Relatedness between two items is determined by the path that connects them, with the highest-level node of the path playing a pivotal role. The distance from the root node to the highest-level node of path 'p' gives another component of relatedness. Nodes close to root node deal with attributes that cause the separation of the entire domain of items into various sub-domains. These attributes tend to be fundamental (or basic) as far as characterization of the domain is concerned, with their values having the ability to segregate and categorize items into various categories/sub-categories. Closer the highest-level node (of path 'p') to the root node, greater is the degree of separation between the items and consequently lower the relatedness. On the other hand, if the highest-level node of path 'p' is at a much lower level, the attribute characterizing the node will be fine-grained and probably specific to the sub-category. The path from the root node to the highest-level node of this path will be longer, with larger number of attribute values identical for the two items under consideration. Thus, the relatedness between the two items increases. We capture this notion by a measure called 'Highest-level relatedness'. The highest-level relatedness [$HR_{A,B}(p)$] for two items connected by path 'p' is defined as the level of the highest-level node [$H_{A,B}(p)$] in path 'p'.

$$HR_{A,B}(p) = \text{level } [H_{A,B}(p)]$$

It is to be noted that the root node is at level 0 and subsequent levels from the root node are labeled by consecutive integers in increasing order starting with 1. Due to this nomenclature of labeling, the relatedness between two items A, B is directly proportional to $HR_{A,B}(p)$. Thus, for the crisp path connecting **knife** and **shoes**, the highest-level relatedness $HR_{knife, shoes}(p) = 1$.

3.3 Node Separation Relatedness [$NSR_{A,B}(p)$]

The length of the path (in terms of nodes) connecting items A and B in a fuzzy taxonomy gives an indication of the conceptual distance between them. Many paths, each of which represents a particular relationship between the two items, can connect two items. Each category node in the path has one of the two items (A, B) either as a 'child' or a 'descendant' node. Only the highest-level node has both items (A and B) as children/descendants nodes. Category nodes other than the highest-level node might consider attributes that are instantiated and specific to one of the two items under consideration. A large number of such distinct attributes specific to each item (resulting in a longer path) reduces relatedness between items. We define a measure of item-relatedness

called node-separation relatedness, $NSR_{a,b}(p)$, which is the length of the simple path [5] 'p' connecting the two items.

$NSR_{A,B}(p)$ = Length of the simple path 'p' connecting nodes A and B.

In Figure 1, node separation relatedness due to the crisp path connecting **knife** and **shoes** is 3 while that due to the fuzzy path connecting them is 5.

4. A Fuzzy Item-Relatedness Measure

In the previous section we had identified three measures that characterize the relatedness between two items A and B represented in a fuzzy taxonomy. These measures viz. highest-level node membership (HM), highest-level relatedness (HR) and node-separation relatedness (NSR) were defined for individual paths connecting two items A and B. Consider the two items **knife** and **shoes** in Figure 1. **Knife** is a crisp member of **household items** while it can also be considered as an **Industrial item** to the extent of 0.5. This arises due to the membership value of **knife** in **cutting tools** being 0.5. Similarly, **shoes** is a crisp member of **household items** while its membership in **industrial items** is 0.6. Thus, from the fuzzy taxonomy, four relationships arise between **knife** and **shoes**. One relationship arises when **knife** and **shoes** both behave or are used as **household items**, the second when both are used in an industrial context. The third and fourth relationships arise when **knife** behaves like a **household item** while **shoes** behave as an **industrial item** and vice versa respectively. Each of the four relationships, represented by a path in the fuzzy taxonomy contributes to a component of relatedness. The total relatedness between the two items is therefore a sum of these four components.

We note that relatedness between items A and B increases if either highest-level node membership (HM) or highest-level relatedness (HR) or both increase. On the other hand, relatedness between two items decreases as the node-separation relatedness measure (NSR) increases in value. Therefore, considering the proportionality of these measures to relatedness, the overall relatedness contributed by the path 'p' can be stated as:

$$OR_{A,B}(p) = \frac{(1 + HR_{A,B}(p))(HM_{A,B}(p))}{NSR_{A,B}(p)}$$

Integer '1' is added to the highest-level relatedness in order to account for the case when root node appears in the path as highest-level node. Total relatedness between A and B is then sum of the relatedness contributed by each path and is given by equation 1. It takes on positive values only.

$$TR(A,B) = \sum_p OR_{A,B}(p) = \sum_p \frac{(1 + HR_{A,B}(p))(HM_{A,B}(p))}{NSR_{A,B}(p)} \quad \text{---(1)}$$

Consider a fuzzy taxonomy having a maximum depth 'k' with the root node being at level 0 (Figure 2). Let us calculate the maximum relatedness that can be contributed by any path in this taxonomy. In order for the relatedness to be maximum here, the node separation relatedness (NSR) should take on the minimum possible value, while the highest-level relatedness measure (HR) should take on the maximum possible value and a 'crisp' path should connect the two items A and B. This condition will be fulfilled only if the path is as shown in Figure 2 (Viz, $A \rightarrow M \rightarrow B$). Here, $NSR_{A,B}(p) = 1$, $HR_{A,B}(p) = k-1$ and $HM_{A,B}(p)$

$= 1$. This gives us $OR_{A, B}(p) = ((k - 1 + 1)/1) \times 1 = k$. This is the upper bound for the relatedness contributed by a single path for a fuzzy taxonomy of depth 'k'.

We note that in a fuzzy taxonomy of depth k, if an item A has memberships in two immediate higher-level nodes, then two paths exist to the root node. Similarly, if item B also has crisp memberships in two immediate higher-level nodes, then a total of $2 \times 2 = 4$ crisp paths exist between the two items. The strength of total relatedness will be highest only if the length of the path is minimum i.e. the two items are siblings, with respect to both their parent nodes (Figure 2).

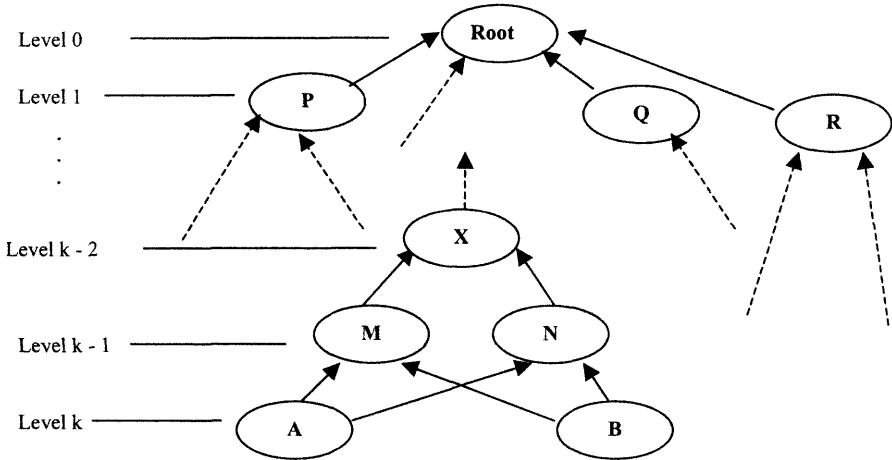


Figure 2: A Fuzzy Taxonomy of level k

There exist two paths of length 1 each passing through the two parent nodes say M and N. Further, there exist two more paths having both the nodes M and N in them and some higher level nodes. The strength of the relationships will be the highest if these two paths are also as short as possible i.e. there is just one node say X that connects nodes M and N. Thus, we have four paths viz. $A \rightarrow M \rightarrow B$, $A \rightarrow N \rightarrow B$, $A \rightarrow M \rightarrow X \rightarrow N \rightarrow B$ and $A \rightarrow N \rightarrow X \rightarrow M \rightarrow B$. The maximum relatedness for two items A and B that have these four crisp paths can be calculated using equation 1. This gives, Total Relatedness (A, B) = $2 \times k + 2(k - 2 + 1)/3 = (2k) + 2(k-1) / 3 = (8k - 2)/3$. Based on this discussion we find that for the fuzzy taxonomy in Figure 1, the maximum theoretical relatedness for any two items A and B, having two paths each to the root node, is 10.

Let us calculate the 'Total relatedness' between **knife** and **shoes** as shown in Figure 1. Four paths exist between them and these are given below. The highest-level nodes in the path are underlined.

Path I: Crisp Path (knife \rightarrow kitchen items \rightarrow household items \rightarrow Footwear \rightarrow shoes.)

Path II: Fuzzy path (Knife \rightarrow cutting tools \rightarrow tools \rightarrow industrial items \rightarrow safety items \rightarrow industrial footwear \rightarrow shoes)

Path III: Fuzzy – Crisp Path

(Knife \rightarrow cutting tools \rightarrow tools \rightarrow Industrial items \rightarrow All Items \rightarrow Household items \rightarrow Footwear \rightarrow Shoes)

Path IV: Crisp – Fuzzy Path

(Knife \rightarrow Kitchen Items \rightarrow Household Items \rightarrow All Items \rightarrow Industrial Items \rightarrow Safety Items \rightarrow Industrial Footwear \rightarrow Shoes)

Details regarding the measures of relatedness are shown in Table 1. Figures in the parentheses represent the membership value of the particular item in the path nodes. The total relatedness existing between **knife** and **shoes** as calculated by the proposed relatedness measure is 1.0503. We derived the maximum possible relatedness between any two items connected by four paths in the fuzzy taxonomy of Figure 2 to be 10. If we compare the two, we see that the relatedness is quite low. This is natural as **knife** and **shoes** are quite different items – varied in their application, purpose and usage. The only point connecting them is that both are **household items** and can be used together as industrial items to a maximum extent of 0.5 and not more.

Table 1: Determination of Total Relatedness between Knife and Shoes

Paths	A=Knife	B=Shoes	NSR _{A, B} (p)	HR _{A, B} (p)	HM _{A, B} (p)	OR _{A, B} (p)
I	Crisp (1.0)	Crisp (1.0)	3	1	1.0	0.667
II	Fuzzy (0.5)	Fuzzy (0.6)	5	1	0.5	0.2
III	Fuzzy (0.5)	Crisp (1.0)	6	0	0.5	0.0833
IV	Crisp (1.0)	Fuzzy (0.6)	6	0	0.6	0.1

5. Discussions

Here we consider a few item-pairs from the fuzzy taxonomy of Figure 1 and discuss the computed results. Table 2 shows the ‘relatedness’ components for five pairs of items from Figure 1. Let us consider the item pair (**shoes** - **safety shoes**). The ‘crisp’ path i.e. path I considers **shoes** as a **household item** and **safety shoes** as an **industrial item** while the fuzzy path (path II) considers the reverse. The relatedness components contributed by these two paths are quite low. This is to be expected as these paths consider the relationships when the two items behave (or are used) as items in different domains viz. household and industrial domains respectively. The fuzzy path contribution is lower than the crisp path contribution as the items cannot be considered as ‘complete’ members in their respective domains. Another interesting observation is with respect to the contribution made by paths III and IV. Path IV considers both **shoes** and **safety shoes** as **footwear items** under **household items**. Although the membership of **safety shoes** in **footwear** is fuzzy (0.4), the fact that the two items are siblings and thus can substitute one another in the same parent domain (household) to a maximum extent of 0.4 strengthens the relatedness between **shoes** and **safety shoes**. This aspect is reflected by a high contribution of 1.2 to the total relatedness measure. Similarly, **safety shoes** and **shoes** when used in industrial domain have a relatedness of 2.4. Intuitively, we know that **Shoes** and **Safety Shoes** are footwear whose primary purpose is to protect the feet of the wearer. Naturally, we would expect a high relatedness between them, except for the fact that they are normally used in quite different settings. Paths I and II however emphasize the fact that they belong to different domains. Failure to consider every path will result in an understatement of the total relatedness.

Consider item pairs 3 and 4 from Table 2. **Spoon** and **Shoes** belong to the same domain viz. **Household Items**. Therefore, the contribution of crisp path I is greater than the contribution of the crisp path for **spoon** and **safety shoes** – items that belong to two different sibling domains. We note that **spoon** and **safety shoes** are related to a greater extent through path IV as compared to path I.

Table 2: A comparison of Item Relatedness for sample item-pairs

Sr. No.	Item A	Item B	Path I (Crisp–Crisp) $OR_{A,B}(I)$	Path II (Fuzzy–Fuzzy) $OR_{A,B}(II)$	Path III (Fuzzy–Crisp) $OR_{A,B}(III)$	Path IV (Crisp–Fuzzy) $OR_{A,B}(IV)$	TR (A, B)
1	Knife	Shoes	0.667	0.2	0.0833	0.1	1.0503
2	Shoes	Safety Shoes	0.1667	0.0667	2.4	1.2	3.8334
3	Spoon	Shoes	0.667	----	----	0.1	0.767
4	Spoon	Safety Shoes	0.1667	----	----	0.2667	0.4334
5	Knife	Industrial Knife	0.1667	0.05	2.0	0.9	3.1167

This is because both are considered as items belonging to the **household** domain, even though the membership value of **safety shoes** in the **household** domain is fuzzy (i.e. 0.4). Item pair {**spoon**, **shoes**} is analogous to pair {**knife**, **shoes**} since **spoon** and **knife** are siblings. However, **spoon** does not have utility as **industrial item** unlike **knife**. This results in only two paths and lower relatedness between **spoon** and **shoes**. Item pair 5 is similar to pair 2 in the sense that the two items in each pair have their primary purpose as the same. However they are members of different domains. We note that **knife** and **industrial knife** are weakly related to each other as compared to **shoes** and **safety shoes**. This is due to lower fuzzy memberships in the case of items **knife** and **industrial knife**.

6. Summary

In this paper we utilized the notion of fuzzy taxonomy similar to the one made use of by Chen et al. [3] in the context of data mining. Using the taxonomy as our basis we introduced three notions of relatedness and then combined them to get a total relatedness measure, “Total relatedness” being inversely related to “interestingness”. With the help of examples, we demonstrated the appropriateness of the measure and intuitiveness of the results.

References:

- [1] A. Silberschatz and A. Tuzhilin, "What makes Patterns Interesting in Knowledge Discovery Systems," in *IEEE Transactions on Knowledge and Data Engineering*, Vol. 8, No. 6, pp 970-974, December, 1996.
- [2] D. K. Subramanian, V. S. Ananthanarayana and M. Narasimha Murty, "Knowledge-Based Association Rule Mining using And-Or Taxonomies," to appear in *Knowledge Based Systems*, 2002.
- [3] G. Chen, G. Wets and K. Vanhoof, "Representation and Discovery of Fuzzy Association Rules (FARs)", Institute of applied Economic Research (ITEO), Limburg University Centre (LUC), Belgium, Research Paper Series, ITEO No: 00/01, March, 2000.
- [4] G. J. Klir and B.Yuan, "Fuzzy Sets and Fuzzy Logic: Theory and Applications," Prentice Hall of India Private Limited, New Delhi, 1997.
- [5] H. J. Hamilton and D. R. Fudger, "Estimating DBLEARN's Potential for Knowledge Discovery in Databases," *Computational Intelligence*, Volume 11, Number 2, pp. 280-296, 1995.
- [6] J.M. de Graaf, W.A. Kusters and J.J.W. Witteman, "Interesting Association Rules in Multiple Taxonomies", presented at BNAIC'00, Kaatsheuvel, November 1/2, 2000 (Proceedings (A. van den Bosch and H. Weigand, editors), pp. 93-100).
- [7] J.M. de Graaf, W.A. Kusters and J.J.W. Witteman, "Interesting Fuzzy Association Rules in Quantitative Databases," proceedings of PKDD 2001 (The 5th European Conference on Principles of Data Mining and Knowledge Discovery), Springer Lecture Notes in Artificial Intelligence 2168 , editors L. De Raedt and A. Siebes, pp. 140-151, Freiburg, Germany, September 3/5, 2001
- [8] N. Deo, "Graph Theory with Applications to Engineering and Computer Science," Prentice Hall of India Private Limited, 1989.
- [9] P. Tan, V. Kumar, and J. Srivastava, "Selecting the Right Interestingness Measure for Association Patterns," accepted for the *Eighth ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD-2002)*, July 23-26, 2002.
- [10] R. Srikant and R. Agrawal, "Mining Generalized Association Rules," in *Proceedings of the 21st VLDB conference*, Zurich, Switzerland, pp 407-419, Sept. 1995.

Factor Analysis with Qualitative Factors as Fuzzy Numbers

Elisabeth RAKUS-ANDERSSON*

*Department of Health, Science and Mathematics, Blekinge Institute of Technology, S-37179
Karlskrona, Sweden, E-mail: Elisabeth.Andersson@bth.se*

Leszek ZAKRZEWSKI

*Systems Research Institute of the Polish Academy of Sciences (PhD student), Warsaw,
Poland, E-mail: L.Zakrzewski@ibspan.waw.pl*

Abstract. The classical crisp version of Factor Analysis is seldom used in the case of qualitative factors by reason of the lack of appropriate level criteria referring to these factors. We now propose a fuzzy interpretation of the method, which gives a possibility to investigate the strength of the factor influence on a tested variable. By assuming that fuzzy numbers in L - R form represent both the variable and the factors, as the qualitative parameters, we are capable of performing all the operations that follow the Factor Analysis algorithm. Even the introduction of the conception proposing a new fuzzy space with particularly defined operations on fuzzy numbers helps to obtain satisfactory results.

1. Introduction

As far as we know the data is collected by means of a questionnaire in many scientific fields. The technical, biological or sociological parameters are thus described as the features characterized by some alternative answers possessing a certain code. Many scientists are interested in finding the knowledge about the substantial influence of chosen factors on a considered variable. Factor Analysis is one of the statistical methods, which solves the sketched problem for quantitative parameters represented by mean values within level groups. To introduce similar values for the qualitative representatives of the groups taking place in Factor Analysis table, we propose fuzzy numbers in L - R representation as the data. In this way we try to avoid counting the sample size n that sometimes negatively affects the expected results. Since the classical operations on fuzzy numbers can extend the spreads of the results in the inconvenient way, we suggest accepting the concept of a number space with members, which can be determined verbally. The operations on fuzzy numbers as well as the problem of fuzzy number comparisons are still subjects of investigations and trials [1, 2, 3, 4, 5, 7, 8, 12]. By making another attempt we have also created our own model adapted to the purpose of applying Factor Analysis with the qualitative data to the appreciation of the variable behavior when observing an action of the factors. By the way, we prove that the subspaces of the fuzzy number space proposed in this dissertation fulfill the conditions to be interpreted as rings, something which is a counterpart of the rules constituting the theoretical base for the sets Z or R .

* Corresponding author

2. Fuzzy Number Space and Operations on Fuzzy Numbers

Some experiments carried out on a sample of objects lead to imprecise values of the observed parameters. The scientists often seek a way of expressing the data when it is not known exactly. The conception of a fuzzy number seems to be useful in the considered case.

Let us recall the conception of a fuzzy number M in L - R representation as the following definition [1, 6, 13].

Definition 1

A fuzzy number M is of L - R type if there exist reference functions L (for left) and R (for right), and scalars $a > 0, \beta > 0$ with

$$\mu_M(z) = \begin{cases} L\left(\frac{m-z}{\alpha}\right) & \text{for } z \leq m \\ R\left(\frac{z-m}{\beta}\right) & \text{for } z \geq m, \end{cases} \quad (1)$$

where m , called the mean value of M , is a real number, and α and β are called the left and right spreads, respectively. Symbolically, M is denoted by (m, α, β) .

Suppose that $M = (m, \alpha, \beta)$ represents the fuzzy number named as, e.g., "close to m " or "very close to m ". We try to designate the difference between "close" and "very close" by establishing the appropriate values of the spreads α and β for both fuzzy numbers. Let us suggest a list containing the most popular, verbal expressions, which are likely used for the imprecise numbers by some scientists. We denote the list by $LIST_m$ and state its contents as [10]

$$\begin{aligned} LIST_m = \{ & M_0 = \text{exactly } m, M_1 = \text{almost exactly } m, M_2 = \text{very close to } m, \\ & M_3 = \text{close to } m, M_4 = \text{rather close to } m, M_5 = \text{appreciatively } m, \\ & M_6 = \text{perhaps } m \}. \end{aligned} \quad (2)$$

Each notion of the $LIST_m$ is proposed to have an L - R representation suggested as

$$M_{d,i} = (m, di, di), m \in \mathbb{Z}, i = 0, 1, 2, 3, 4, 5, 6, d \in \mathbb{R}, \quad (3)$$

where d is a fix multiplier accommodating the breadth of the spreads di to a concrete problem. The assumption $m \in \mathbb{Z}$ seems to fit best in the case of imprecise numbers instead of $m \in \mathbb{R}$. If m is a real number it can be given as the number with a very high decimal precision, and we probably will not need any fuzzy representation of such m .

By accepting that $L(z) = R(z) = -z + 1$ we derive new formulas of the membership functions for $M_{d,i}$, $i = 1, \dots, 6$, expanded as

$$\mu_{M_{d,i}}(z) = \begin{cases} L\left(\frac{m-z}{di}\right) = -\frac{m-z}{di} + 1 = \frac{z+di-m}{di} & \text{for } z \in [m-di, m] \\ R\left(\frac{z-m}{di}\right) = -\frac{z-m}{di} + 1 = \frac{di+m-z}{di} & \text{for } z \in [m, m+di] \end{cases} \quad (4)$$

If we use the fuzzy numbers $M_{d,i}$, $i = 0, 1, 2, \dots, 6$, defined by (2), (3) and (4) as the data in the operations of addition, subtraction, multiplication and division we would like to cause the appearance of a result y that yields the value resembling an arbitrary value x belonging to $LIST_m$. We expect that y should be included in the class of numbers specified by (2). By performing the operations on fuzzy numbers due to [1, 6, 13] or [2, 3] we recognize that we never get the result y with the spreads satisfying the condition (3). To demonstrate this assertion let us add "very close to 5" to "appreciatively 3", $d=1$, when using the classical definition of addition of two fuzzy numbers [1]. Hence, $(5, 2, 2)+(3, 5, 5)=(8, 7, 7)$. The result of the addition is the number with large spreads that is not a member of $LIST_{m=8}$, either. The multiplication of these numbers will bring the effect $(5, 2, 2)\times(3, 5, 5)=(15, 5\times 5+3\times 2, 5\times 5+3\times 2)=(15, 31, 31)$ as the very imprecise number with the huge values of a and β despite of the fact that neither $(5, 2, 2)$ nor $(3, 5, 5)$ are spread so widely. Moreover, we are not sure how to express verbally such a number. Even testing the operations, recommended by [2, 3], we get the result $(5, 2, 2)\times(3, 5, 5)=(5\times 3, \max(2\times 3, 5\times 5), \max(2\times 3, 5\times 5))=(15, 25, 25)$ which still preserves the large values of the spreads, and we note that both a and β also will exceed the borders of the numbers placed in (3).

Let us consider the space of fuzzy numbers of the type $M_{d,i}=(m, di, di)$, $m \in Z$, $i = 0, 1, 2, 3, 4, 5, 6$, $d \in R$ denoted by $FN(LIST)$. By performing the operations of addition, subtraction, multiplication and division we wish to obtain other numbers looking like $M_{d,i}$. We thus propose the following operations on the fuzzy numbers from $FN(LIST)$ by defining them as [10]

$$\begin{aligned}
 M_{d,i} + N_{d,j} &= (m, di, di) + (n, dj, dj) = (m + n, \max(di, dj), \max(di, dj)) \\
 M_{d,i} - N_{d,j} &= (m, di, di) - (n, dj, dj) = (m - n, \max(di, dj), \max(di, dj)) \\
 M_{d,i} \times N_{d,j} &= (m, di, di) \times (n, dj, dj) = (mn, \min(di, dj), \min(di, dj)) \\
 M_{d,i} \div N_{d,j} &= (m, di, di) \div (n, dj, dj) = (m \div n, \min(di, dj), \min(di, dj))
 \end{aligned} \tag{5}$$

for $m, n \in Z$, $i, j = 0, 1, 2, 3, 4, 5, 6$, $d \in R$.

These operations, which intuitively are interpreted as similar to the operations occurring in the case of the largest t -norm for multiplication (min) and the smallest s -norm for addition (max) [13], warrant that their results always will be the fuzzy numbers from (2). Let us note that the list of $M_{d,i}$ has been created according to the scientists' advice (especially physicians). Hence, the users who define the entries of data in the way that is comfortable for them - in other words by choosing any description belonging to $LIST_m$, $m \in Z$, - expect that they obtain a result y as another fuzzy number interpreted as one of the expressions from, e.g., $LIST_n$, $n \in Z$.

If both d and i are fix numbers then the set of fuzzy numbers of the type (m, p, p) , $m \in Z$, $p=di$ is a constant, will possess all the properties characterizing a ring.

Theorem 1

The set of fuzzy numbers $FN(z, p) = \{(z, p, p) : z \in Z, p = id \text{ is fix}\} \subset FN(LIST)$ with the operations of addition and multiplication defined by (5) is a ring.

Proof

$FN(z, p)$ is a commutative group with respect to addition of fuzzy numbers in accordance to (5) since the following conditions are satisfied:

- 1) For any $(m, p, p) \in FN(z, p)$ and $(n, p, p) \in FN(z, p)$,
 $(m, p, p) + (n, p, p) = (m + n, \max(p, p), \max(p, p)) = (m + n, p, p) \in FN(z, p)$ - the closure of $FN(z, p)$.
- 2) For any $(m, p, p) \in FN(z, p)$, $(n, p, p) \in FN(z, p)$ and $(s, p, p) \in FN(z, p)$,
 $((m, p, p) + (n, p, p)) + (s, p, p) = (m, p, p) + ((n, p, p) + (s, p, p))$ - associativity.
- 3) There is an element $(0, p, p) \in FN(z, p)$ ($0 \in Z$) such that
 $(m, p, p) + (0, p, p) = (0, p, p) + (m, p, p) = (m, p, p)$ for all $(m, p, p) \in FN(z, p)$ - identity.
- 4) For all $(m, p, p) \in FN(z, p)$ there is $(-m, p, p) \in FN(z, p)$, $(-m \in Z)$ satisfying
 $(m, p, p) + (-m, p, p) = (0, p, p)$ - inverse.
- 5) For any $(m, p, p) \in FN(z, p)$ and $(n, p, p) \in FN(z, p)$, $(m, p, p) + (n, p, p) = (n, p, p) + (m, p, p)$ - commutativity.

The set $FN(z, p)$ with multiplication fulfils conditions 1)-3) according to the following statements:

- 1) For any $(m, p, p) \in FN(z, p)$ and $(n, p, p) \in FN(z, p)$, $(m, p, p) \times (n, p, p) = (mn, \min(p, p), \min(p, p)) = (mn, p, p) \in FN(z, p)$ - the closure of $FN(z, p)$.
- 2) For any $(m, p, p) \in FN(z, p)$, $(n, p, p) \in FN(z, p)$ and $(s, p, p) \in FN(z, p)$,
 $((m, p, p) \times (n, p, p)) \times (s, p, p) = (m, p, p) \times ((n, p, p) \times (s, p, p))$ - associativity.
- 3) There is an element $(1, p, p) \in FN(z, p)$ ($1 \in Z$) such that
 $(m, p, p) \times (1, p, p) = (1, p, p) \times (m, p, p) = (m, p, p)$ for all $(m, p, p) \in FN(z, p)$ - identity.

Finally we check the reliability of the distributive law:

$$(m, p, p) \times ((n, p, p) + (s, p, p)) = ((m, p, p) \times (n, p, p)) + ((m, p, p) \times (s, p, p)).$$

Since the set $FN(z, p)$ satisfies all the axioms above, it will be interpreted as the ring. Let us note that $FN(z, p)$ becomes the set Z (also the ring) if $i=0$. It would be desirable to find in the nearest future such operations, performed on fuzzy numbers from the entire set $FN(LIST)$, which allow it to be the ring.

3. Basic Assumptions of Factor Analysis

By carrying out an experiment and collecting data from it we try to examine the changes in an observed variable that depends on certain factors. To make the process simpler let us suppose that the variable X is dependent on three factors A, B and C , each at two levels. We usually think of these levels as the "low" and "high" levels of the factor. This design is a 2^3 factorial design, and it has eight influence combinations. We accept the following annotation pattern to denote different combinations of the factor influence, remembering that lowercase letters are used for the results on the higher levels and the lack of symbols refers to the results connected to the lower levels. The symbol (1) is reserved for a result of all the factors on the lower levels [9, 11].

Hence, we obtain the following notions with explanations:

(1) - ABC -low,	a - A -high, BC -low,	b - B -high, AC -low,
ab - AB -high, C -low,	c - C -high, AB -low,	ac - AC -high, B -low,
bc - BC -high, A -low,	abc - ABC -high,	

which are involved in the estimation of the main effects of every combination of the factor levels.

The data concerning the values of X is divided into the eight groups, which preserve the factor levels. Afterwards we evaluate the representative values of X for each group and present the results from the experiment in the following table.

Table 1: The group values of the variable X

A -low				A -high			
B -low		B -high		B -low		B -high	
C -low	C -high	C -low	C -high	C -low	C -high	C -low	C -high
(1)	c	b	bc	a	ac	ab	abc

The total effect of the main factor A is computed as the difference between the results of X on the higher level of A (all symbols, which contain a) and the results on the lower level of A (all symbols that do not contain a). In the similar way we appreciate the total effects of B and C . To evaluate the influence of interactions of two factors, e.g., AB we create the difference between the effects on the same level of A and B (either ab appears or ab is lacking) and the effects on different levels of A and B (either a or b appears). The common effect of A , B and C is the average difference between AB interactions at the two levels of C .

The total effects of the factor influence are given in accordance with the following formulas:

$$\begin{aligned}
 A &= (a + ab + ac + abc) - ((1) + b + c + bc) \\
 B &= (b + ab + bc + abc) - ((1) + a + c + ac) \\
 C &= (c + ac + bc + abc) - ((1) + a + b + ab) \\
 AB &= ((1) + c + ab + abc) - (a + b + ac + bc) \\
 AC &= ((1) + b + ac + abc) - (a + c + ab + bc) \\
 BC &= ((1) + a + bc + abc) - (b + c + ab + ac) \\
 ABC &= (a + b + c + abc) - ((1) + ab + ac + bc).
 \end{aligned} \tag{6}$$

The pattern below explains how to calculate the total effect values of different interactions of factor levels after stating the results of X in the next table. The third, the fourth and the fifth column contain sums and differences performed on the numbers coming from the previous columns. The sums consist of two terms belonging to two adjacent rows while the differences are calculated between the values from even and odd rows. It is easy to note that the last, fifth column comprises the values of the total effects, which are equal to (6).

Table 2: The total influence of factor interactions on the values of X

Experiment	Results	First sum and difference	Second sum and difference	Total effect of factors
ABC -low	(1)	(1)+ a	((1)+ a)+(b + ab)	-
A -high, BC -low,	a	b + ab	(c + ac)+(bc + abc)	(a -(1))+(ab - b)+(ac - c)+(abc - bc)
B -high, AC -low	b	c + ac	(a -(1))+(ab - b)	(b + ab)-((1)+ a)+(bc + abc)-(c + ac)
AB -high, C -low	ab	bc + abc	(ac - c)+(abc - bc)	(ab - b)-(a -(1))+(abc - bc)-(ac - c)
C -high, AB -low	c	a -(1)	(b + ab)-((1)+ a)	((c + ac)+(bc + abc))-((1)+ a)+(b + ab))
AC -high, B -low	ac	ab - b	(bc + abc)-(c + ac)	((ac - c)+(abc - bc))-((a -(1))+(ab - b))
BC -high, A -low	bc	ac - c	(ab - b)-(a -(1))	((bc + abc)-(c + ac))-((b + ab)-((1)+ a))
ABC -high	abc	abc - bc	(abc - bc)-(ac - c)	((abc - bc)-(ac - c))-((ab - b)-(a -(1)))

By placing the values included into the last column in ascending order (we care of their magnitudes only), we grade the factors that affect X .

To accomplish the upgrading of the factors we insert the rule: "the larger value the greater influence". Even the sign can be interpreted as an indication for the direction of changes in the variable X after the factors' action. The negative sign demonstrates a tendency to taking lower values by X while the positive sign assures that the values of X will grow when essential factors participate in the experiment.

4. Fuzzification of Factor Analysis

Factor Analysis, described in the classical version in Section 3, is usually applied to a variable and some factors, which are the measurable parameters. The method entails the special adaptation to fuzzy conditions in the case of qualitative data.

Let us begin with the factors A, B, C that are the features explored as the data by means of a questionnaire. Suppose that each of them is designed as a list of alternative answers to a certain question. The answers are numbered from 1 to k and have the same tendency to be stated in the order from the most negative to the most positive with respect to their influence on the positive development of the variable X . Let us interpret the list of answers for an arbitrary factor F as a fuzzy set $F = \{1, \dots, k\}$, $k \in N$ with the membership function

$$\mu_F(y) = \frac{y-1}{k-1} \text{ for } y = 1, \dots, k. \quad (7)$$

We point out that the most negative answer has the value of the membership function equal to 0 while the most positive one corresponds to the value of 1, which fits to our intuitive insight of the solution desired.

To classify the answer y , $y=1, \dots, k$, within either the "low" or "high" level of F we prove the criterion associated with the value of the membership grade of y . If $\mu_F(y) \leq 0.5$ the answer y is reckoned in the lower level of F , otherwise it belongs to F 's higher level.

Suppose further that the variable X is also measured in units among 1 and v , $X = \{1, \dots, v\}$, $v \in N$, ranked in the ascending order with respect to the positive development and the importance of X . We examine a sample with the codes of X , when taking care of the A, B, C factor levels in accordance with (7), and divide the codes into eight groups as it has been revealed in Section 3. If we compute an arithmetic mean for each group we certainly will obtain a real value, which does not belong to $X = \{1, \dots, v\}$, $v \in N$.

By researching into some samples we can appreciate the representatives for the groups as fuzzy numbers of the type (2) in the common form

$$\text{"representative of } X \text{ for factor groups"} = (m, di, di), m \in \{1, \dots, v\}, i = 0, 1, \dots, 6. \quad (8)$$

If the space between $x = 1, \dots, v$ are equal to 1 then it will be advised to set $d=0.1$.

We can now express each mean of the group as the fuzzy number, e.g., "very close to 2" or "perhaps 2" and replace it by the associated fuzzy number according to (2) and (8). In this way we have avoided showing consideration for the sample size and different results coming from different samples.

By placing the evaluated means for the group as fuzzy numbers (8) in Table 2 and performing the operations on fuzzy numbers in accordance with (5) we get the contents in

the fifth column as a set of fuzzy numbers $\Theta = \{(\omega, di, di)\}, \omega \in Z$, which gathers the knowledge about the total effects of the factor influence on the variable X .

Let us also denote the set of all $|\omega|$ that constitute the absolute mean values in Θ , by $\Theta(\omega)$, where $\Theta(\omega) = \{|\omega| : \omega \text{ is the mean of } (\omega, di, di) \in \Theta\}$. We reorder the mean values (in magnitude) of the fuzzy numbers from the last column in the ascending order to obtain a support of a fuzzy set "decision" = $\left\{ \min_{|\omega| \in \Theta(\omega)} (|\omega|), \dots, \max_{|\omega| \in \Theta(\omega)} (|\omega|) \right\}$ with a membership function that is found as the formula

$$\mu_{\text{decision}}(|\omega|) = \frac{|\omega| - \min_{|\omega| \in \Theta(\omega)} (|\omega|)}{\max_{|\omega| \in \Theta(\omega)} (|\omega|) - \min_{|\omega| \in \Theta(\omega)} (|\omega|)}. \quad (9)$$

In order to determine which factor interactions are essential we only accept these combinations that have total effects expressed by $(\omega, di, di) \in \Theta$ with the membership degrees of $|\omega| \in \Theta(\omega)$ greater than 0.5. The degrees are computed by using of (9). The last constraint replaces the classical criterion based on the Fisher test, which cannot be useful anymore.

5. Example

By means of the questionnaire that was involved in an inquiring process of the computer science education, we would like to infer about the influence of different factors on the effectiveness of learning.

In order to investigate the problem carefully some researches were carried out among the beginners studying at different colleges and faculties. They were taught and instructed in the computer science at their schools. The efficiency of the instructions in the computer science via the estimation of the schoolboys' skills at informatics is the main purpose of the analysis below.

The qualitative variable X = "the estimation of the skills in informatics" is thus considered and measured in the scale from 1 to 6 (as school marks). The variable X is dependent on three factors on two levels (lower and higher).

X was associated with the following factors named as: A = "knowledge and skills of the teacher", B = "the quality of schoolbooks", C = "the own contribution in learning".

These factors are also the qualitative features expressed in the scale with the codes 1, ..., k , where k is a number of alternative answers. The answers are graded up in the direction from the most negative to the most affirmative when paying attention to their action and influence in the case of X .

The factor A = "knowledge and skills of the teacher" has $k=5$ and the scale: $A = \{\text{very weak, weak, average, good, as good as gold}\}$.

The k -value for the factor B = "the quality of the schoolbook" is estimated as $k=5$, and the list of answers is established as: $B = \{\text{very bad, bad, medium, good, very good}\}$.

The last factor C = "the own contribution in learning" has the number of alternatives equal to $k=6$, and the contents of upgraded possible expressions is proposed as $C = \{\text{almost none, little, medium, rather large, large, very large}\}$.

The analysis of X was accomplished for three samples (30 members each) containing the schoolboys' answers to the questions concerning the factors A , B and C .

To find the borders between the lower and the higher level of the factors considered, we regarded them as fuzzy sets with the membership function (7).

Such a point of view brought a possibility to compare the membership grades of the codes with the value of 0.5 as it was explained in Section 4. Moreover, the results of X for each sample were divided into eight groups in the manner discussed in Section 3 and 4.

From the first sample we counted the following representatives of X within the groups taking place in the Factor Analysis description: (1) – 3, $a - 3$, $b - 2$, $ab - 4$, $c - 4$, $ac - 4$, $bc - 5$, $abc - 6$.

The second sample gave the results of X stated as (1) – 2, $a - 3$, $b - 2$, $ab - 6$, $c - 4$, $ac - 6$, $bc - 5$, $abc - 6$.

In the third collection of investigated objects we found some material allowing us to appreciate values of X as follows: (1) – 2, $a - 4$, $b - 3$, $ab - 4$, $c - 4$, $ac - 6$, $bc - 6$, $abc - 6$.

In compliance with suggestions made in Section 2 we proposed the fuzzy numbers with corresponding verbal presentations, which could be used as the generalization of data in Factor Analysis groups with respect to the factor levels. The sample effects were carefully observed for they grounded the base of the estimations. The generalized values in Table 1 were assumed to be taken from the list introduced for $d=0.1$ and created below as:

($m, 0, 0$)="exactly m ", ($m, 0.1, 0.1$)="nearly exactly m ",
 ($m, 0.2, 0.2$)="very close to m ", ($m, 0.3, 0.3$)="close to m ",
 ($m, 0.4, 0.4$)="rather close to m ", ($m, 0.5, 0.5$)="appreciatively m ",
 ($m, 0.6, 0.6$)="perhaps m ".

The representing values of X for every group replaced the theoretical values that were written into Table 1. They were placed in the last row of a table, which had the same pattern as Table 1.

Table 3: The group values of the variable X

A-low				A-high			
B-low		B-high		B-low		B-high	
C-low	C-high	C-low	C-high	C-low	C-high	C-low	C-high
"perhaps 2"= (2,0,6,0,6)	"exactly 4"= (4,0,0)	"appreciatively 2"= (2,0,5,0,5)	"close to 5"= (5,0,3,0,3)	"close to 3"= (3,0,3,0,3)	"very close to 5"= (5,0,2,0,2)	"appreciatively 4"= (4,0,5,0,5)	"close to 6"= (6,0,3,0,3)

It is worth emphasizing that the estimates of the representing values for each factor group were guessed without taking into consideration the sample size n .

Then we finally calculated the total influences looking at (6). The operations on the fuzzy numbers were performed with reference to (5). The obtained final effects are shown in the next table.

Table 4: The total influence of factor interactions on the values of X

Experiment	Results	First sum and difference	Second sum and difference	Total effect of factors
ABC-low	(2, 0.6, 0.6)	(5, 0.6, 0.6)	(11, 0.6, 0.6)	-
A-high, BC-low,	(3, 0.3, 0.3)	(6, 0.5, 0.5)	(20, 0.3, 0.3)	(5, 0.6, 0.6)
B-high, AC-low	(2, 0.5, 0.5)	(9, 0.2, 0.2)	(3, 0.6, 0.6)	(3, 0.6, 0.6)
AB-high, C-low	(4, 0.5, 0.5)	(11, 0.3, 0.3)	(2, 0.3, 0.3)	(1, 0.6, 0.6)
C-high, AB-low	(4, 0, 0)	(1, 0.6, 0.6)	(1, 0.6, 0.6)	(9, 0.6, 0.6)
AC-high, B-low	(5, 0.2, 0.2)	(2, 0.5, 0.5)	(2, 0.3, 0.3)	(-1, 0.6, 0.6)
BC-high, A-low	(5, 0.3, 0.3)	(1, 0.2, 0.2)	(1, 0.6, 0.6)	(1, 0.6, 0.6)
ABC-high	(6, 0.3, 0.3)	(1, 0.3, 0.3)	(0, 0.3, 0.3)	(-1, 0.6, 0.6)

The mean values of the results from the fifth column in Table 4 constituted the elements of the fuzzy set "decision" (see Section 4).

The members of the support of „decision”, as the absolute values in the increasing order, found their places in a sequence of integers. In accordance with (9) we could establish the solutions, that implemented the final conclusions, as the membership grades for the entire support of the set "decision". We sum up the answers in the last table.

Table 5: The values of the membership function of the fuzzy set "decision"

Experiment	The mean value $ \omega $ of the total effect	Membership degree of $ \omega $ in the set "decision"
<i>BC</i> -high, <i>A</i> -low (<i>bc</i>)	1	0
<i>AC</i> -high, <i>B</i> -low (<i>ac</i>)	1	0
<i>ABC</i> -high (<i>abc</i>)	1	0
<i>AB</i> -high, <i>C</i> -low (<i>ab</i>)	1	0
<i>B</i> -high, <i>AC</i> -low (<i>b</i>)	3	0.25
<i>A</i> -high, <i>BC</i> -low, (<i>a</i>)	5	0.5
<i>C</i> -high, <i>AB</i> -low (<i>c</i>)	9	1

As the essential combinations of the factors we only accept the ones, which possess the value of the membership grade in the set "decision" greater than 0.5.

As we can see, the greatest importance of the factor influence on the pupils' abilities in informatics is associated with C = "the own contribution of the work". The factor A = "knowledge and skills of the teacher" also affects the affirmative values of X in the substantial grade.

6. Conclusions

The mathematicians who work on some applications of mathematics to other subjects should respect other scientists' opinions; especially if they have the different abilities in formulating their problems in the way they understand best. It means that researchers, non-professional mathematically, often use the verbal expressions to describe a value they need. Being aware of it we have made a trial (confirmed by suggestions in (10)) to introduce a new fuzzy space with fuzzy numbers in the particular L - R form.

It makes possible to choose values required by using simple linguistic formulations and replacing them by appropriate fuzzy numbers from the space. The operations, which are defined in the space, keep the results within the same space without enlarging the borders of the numbers. There exist some subspaces in the space of fuzzy numbers that are rings. The fact of discovering such subspaces reinforces the efforts to seek a total space of fuzzy numbers, which fulfils the conditions of the ring, maybe even the field.

The qualitative data coming from the Factor Analysis experiment could be considered, since new criteria different from the classical ones, were found and adapted to the performance of both the fuzzy numbers and the operations on them.

7. Prognoses

It would be desirable to discover the operations on fuzzy numbers over the total fuzzy space $FN(LIST)$, which convert the space into a ring. We could regard such a space as a counterpart of the sets Z and R .

We also wish to prove the operations that do not lead to dominant values of the spreads in the numbers obtained as results. Unfortunately, both maximum and minimum operations have the tendency to the creation of sharp effects. We will try to invent other operations in the fuzzy space proposed, which should improve the theoretical and practical assumptions.

Acknowledgements

The first author acknowledges The Swedish Royal Academy of Sciences for the grant received in 2002 and allocated for development of Fuzzy Mathematics in applied branches.

References

- [1] D. Dubois and H. Prade, Fuzzy Real Algebra. Some Results, *Fuzzy Sets and Systems* **2** (1978) 327-348.
- [2] Dug Hun Hong, Sungho Lee and Hae Young Do, Fuzzy Linear Regression Analysis for Fuzzy Input-Output Data Using Shape-Preserving Operations, *Fuzzy Sets and Systems* **122**(3) (2001) 513-526.
- [3] Dug Hun Hong, Shape Preserving Multiplications of Fuzzy Numbers, *Fuzzy Sets and Systems* **123** (2001) 81-84.
- [4] R. Goetschel and W. Voxman, Elementary Fuzzy Calculus, *Fuzzy Sets and Systems* **18** (1986) 31-34.
- [5] M. G. Iskander, Comparison of Fuzzy Numbers Using Possibility Programming. Comments and New Concepts, *Computers & Mathematics with Applications* **43** (2002) 833-840.
- [6] J. Kacprzyk, Fuzzy Sets in System Analysis. Polish Scientific Publishing House, Warsaw, 1986 (in Polish).
- [7] Liang-Hsuan Chen and Hai-Wen Lu, An Approximate Approach for Ranking Fuzzy Numbers Based on Left and Right Dominance, *Computers & Mathematics with Applications* **41** (2001) 1589-1602.
- [8] B. Matarazzo and G. Munda, New Approaches for the Comparison of *L-R* Fuzzy Numbers: a Theoretical and Operational Analysis, *Fuzzy Sets and Systems* **118** (2001) 407-418.
- [9] T. Gerstenkorn and E. Rakus, The Application of Factor Analysis in Medical Experiment, *People. Population, Environment* **32** (1989) 156-164 (in Polish).
- [10] E. Rakus-Andersson, The Newton Interpolation Method with Fuzzy Numbers as the Entries, to be published in proceedings of ICNNSC'2002, Springer-verlag – Soft Computing series.
- [11] W. Volk, Applied Statistics for Engineers. McGraw-Hill, Inc., 1969.
- [12] X. Wang and E. E. Kerre, Reasonable Properties for the Ordering of Fuzzy Quantities, (I), (II), *Fuzzy Sets and Systems* **118** (2001) 375-385, 387-405.
- [13] H. J. Zimmermann, Fuzzy Set Theory and Its Applications, 3^d Edition. Kluwer Academic Publish, Boston, 1996.

An Imperfect String Matching Experience Using Deformed Fuzzy Automata

J.J. Astrain J.R. Garitagoitia J. Villadangos F. Fariña A. Córdoba

J.R. González de Mendivil

Dept. Matemática e Informática

Universidad Pública de Navarra

Campus de Arrosadía, 31006 Pamplona, Spain

{josej.astrain, joserra, jesusv, fitxi, alberto.cordoba, mendivil}@unavarra.es

Abstract. This paper presents a string matching experience using deformed fuzzy automata for the recognition of imperfect strings. We propose an algorithm based on a deformed fuzzy automaton that calculates a similarity value between strings having a non-limited number of edition errors. Different selections of the fuzzy operators for computing the deformed fuzzy automaton transitions allows to obtain different string similarity definitions. The selection of the parameters determining the deformed fuzzy automaton behavior is obtained via genetic algorithms.

1 INTRODUCTION

The applications of pattern recognition based on structural and syntactic methods have to cope with the problem of recognizing strings of symbols that do not fit with any one of the defined patterns. The problem is usually resolved by defining a function that allows measuring the similarity (distance or dissimilarity) between pairs of strings [14].

This approach consists basically in the construction of a dictionary by using the representatives of the classes. For the recognition of an erroneous string, the string is compared with every sample in the dictionary. A similarity value (or distance) is then computed with every one of the samples in terms of the edit operations (normally insertion, deletion and substitution of a symbol) needed for the transformation of a string into another one. Finally, the string is classified in the class whose representative obtains greater similarity (less distance). In the literature different definitions for string distance, as well as algorithms that calculate them, are proposed [9, 10, 1, 12, 15].

Recognition rates of pattern classification systems are sometimes low because different steps in the recognition process are considered in isolated and sequential way. Each step makes a decision at its own level, and it could be erroneous. Later steps have to do their task taking into account erroneous data, which invariably affects the recognition rate. For instance, text recognizers usually contain steps for segmentation, isolated character recognition, word recognition, sentence recognition, and so on, which act sequentially in such a way that for the classification of a character it is not taken into account the word in which the character is located. Thus, in the work of a particular step, the information level pertaining to a later step is not considered.

In order to improve recognition rates and, at the same time, to keep the work separate in different steps, the use of fuzzy symbols has been proposed [3, 2, 7, 16] as an adequate way to represent the ambiguity in the classification of previous steps. A fuzzy symbol is a fuzzy set defined over the total set of symbols, with its membership function representing the similarity between the observed symbol and every symbol in the set. Within the concept of string matching, the problem of classifying strings presents now a higher complexity because it is necessary to work with strings containing fuzzy symbols which, in addition to edition errors, present imprecise information in the fuzzy symbols.

We propose a fuzzy automaton for the classification of strings containing any amount of errors, that can use different fuzzy operations (t -conorms and t -norms) [7] implementing different distance (in fact, similarity) definitions. Deforming the fuzzy automaton [11], it can be performed a string matching accepting as inputs imperfect strings of fuzzy symbols. Given a particular problem we have to choose adequately the fuzzy values of the transitions as well as the values of the t -conorm/ t -norm parameters. The problem of selecting such parameters is an optimization problem. In this paper, we propose tuning the parameters of the fuzzy automaton using a genetic algorithm [5].

In order to validate our method, an experimental system for the recognition of texts has been developed. The character classifiers provide fuzzy characters that permit to represent the ambiguity of the results obtained by the classifiers of isolated characters [17, 3, 2]. The contextual processing step of the system makes the comparison between the strings of fuzzy characters and the strings of a dictionary representing the lexicographical context. This step is implemented by the proposed deformed fuzzy automaton. The experimental results show that the deformed system has a great capacity for recovering errors. This fact leads to high word recognition rates even in the presence of a high amount of errors.

The rest of the paper is organized as follows. In Section 2 the algorithm of the deformed fuzzy automaton that computes similarity between strings is introduced. Section 3 is devoted to show experimental results obtained when the deformed fuzzy automaton is applied to a problem of text recognition. Section 4 presents the conclusions. Finally, references end the paper.

2 STRING SIMILARITY BASED ON DEFORMED FUZZY AUTOMATA

This section is devoted to the introduction of a deformed fuzzy automaton that computes string similarity between two strings, the *observed* and the *pattern* strings respectively.

Initially, a finite deterministic automaton $M(\omega)$ which recognizes the pattern string ω is defined. This automaton will not accept the observed string α unless $\alpha = \omega$. Second, this automaton is modified as it was introduced in [4], obtaining a fuzzy automaton $MF(\omega)$ in order to accept every observed string α providing a value for the similarity between α and ω . This modification includes the management of all the possible edit operations (insertion, deletion, and change of a symbol).

When the symbols of an observed string are processed by a isolated character classifier (ICC), a fuzzy symbol representing the ambiguity of each symbol classification is obtained. The value of this fuzzy symbol represents the degree of proximity of the pattern symbol to the observed symbol. Then, it is necessary to work with imperfect strings of fuzzy symbols, that is, strings which may contain insertion, deletion, and substitution of fuzzy symbols (edition errors).

Third, in order to work with imperfect fuzzy symbols, the fuzzy automaton is deformed as it was explained in [4]. Then, an algorithm for the computation of the deformed fuzzy automaton is introduced (see figure 1).

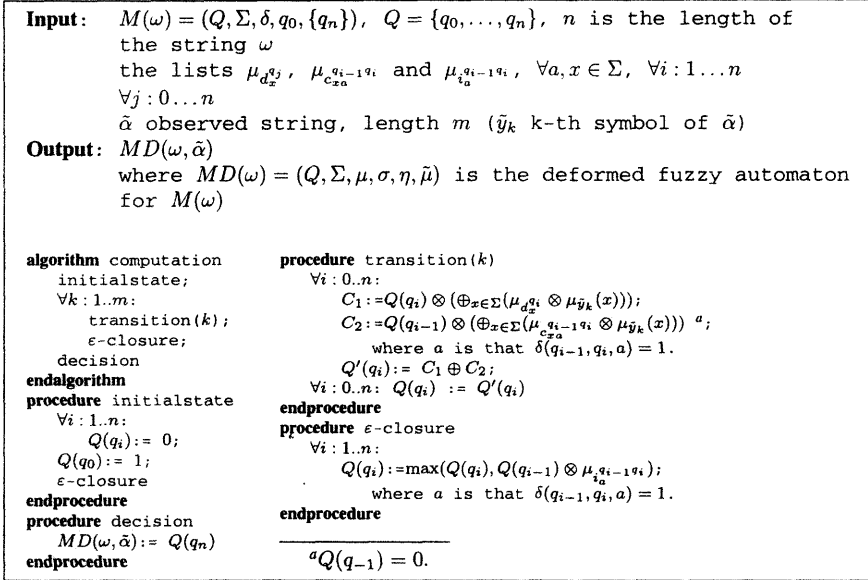


Figure 1: Algorithm for the deformed fuzzy automaton.

The main program is given by the algorithm computation. The algorithm starts by building the initial fuzzy state which is given by σ (from the definition of $MD(\omega)$). The body of the algorithm is a loop in which the procedures transition and ϵ -closure are executed for every fuzzy symbol \tilde{y}_k of the observed string $\tilde{\alpha}$. These two procedures compute the state transition by a fuzzy symbol and the empty string, respectively. Finally, the procedure decision, computes the final fuzzy state given by η . There are only two possible transitions to reach state q_i : One from state q_{i-1} , representing a substitute operation (C_2 in figure 1), the other one from the same state q_i , representing a delete operation (C_1 in figure 1). The procedure ϵ -closure represents insert operations.

Operators \otimes and \oplus represent the t -norm and the t -conorm respectively. One can note that for each replacement of a t -conorm and a t -norm, a different fuzzy automaton is obtained, and thus, a different similarity operator. In [4] we show that when the operators \oplus and \otimes are replaced by the *maximum* and the *algebraic product* operators respectively, a similarity operator that calculates (transforming the output value as $-\log(MF(\omega, \alpha))$) the Generalized Levenshtein Distance [9] is obtained. The use of different t -conorms and t -norms may be of interest for practical problems due to different aggregations they provide.

One can note that the string similarity provided by our deformed fuzzy automaton has a more flexible behavior because it makes use of a parametric t -norm/ t -conorm for computing the automata transitions. In the following, we use the Hamacher t -norm/ t -conorm and we denote γ to the parameter that Hamacher uses to cover the space ($\gamma \geq 0$) of possible t -conorms and t -norms [7].

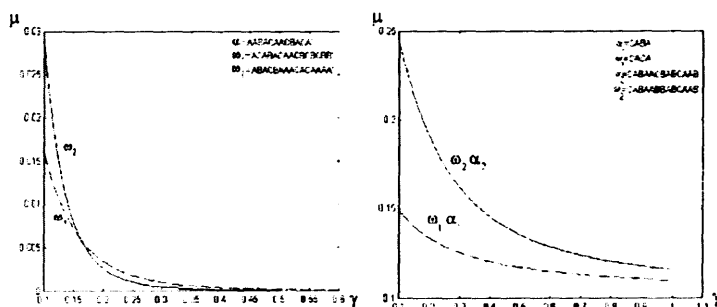


Figure 2: (Left) Effects of changing the γ parameter of the Hamacher t -norm/ t -conorm when comparing the observed string α with two pattern strings ω_1 and ω_2 . (Right) Avoiding the effects of the string length.

In the figure 2 (left), we show the effects of changing the γ parameter of the Hamacher t -norm/ t -conorm when comparing the observed string α with two pattern strings ω_1 and ω_2 varying the value of the Hamacher's parameter. The figure shows that the final decision depends not only in the number of edit operations to transform α into ω_1 or ω_2 but in the similarity measure obtained varying the t -norm/ t -conorm. In fact, we could select the t -norm/ t -conorm that gives the desired result.

The Levenshtein Edit Distance is dependant of the length of the strings. This fact has motivated to researches to introduce Normalized Edit Distances [10], that consider the relation between the number of errors and the length of the strings when their similarity is analyzed. A similar effect can be obtained with our deformed fuzzy automaton by using different values of the t -norm/ t -conorm parameter for each target string length. Consider the example depicted by figure 2 (right). It shows the results obtained for different values of Hamacher parameters when recognizing a short input string, α_1 , with one substitution error and a long input string α_2 also with one substitution error. If we need that the errors in short strings be more significant than in long strings, we could assign to the short string a value of the Hamacher's parameter higher than the one to the long string. For example, selecting $\gamma = 1$ for computing the fuzzy automaton of ω_1 while $\gamma = 0,2$ when computing the fuzzy automaton of ω_2 , we will obtain the desired result. The similarity between α_2 and ω_2 is higher than the similarity between α_1 and ω_1 .

As conclusion, the membership values of the elementary edit operations and the value of the parameter of the selected parametric t -norm/ t -conorm for each pattern string as the parameters of the system must to be considered. Those parameters must to be tuned in order to adjust the system to each concrete problem.

3 EXPERIMENTAL RESULTS

In order to evaluate the proposed method, an experimental system for hand-printed text recognition has been considered. In the experimental system, the contextual post processing [6] is implemented by the related deformed fuzzy automaton. The objective is to analyze the capability of the deformed system to correct the edition errors that are introduced during text creation (typographical and lexicographical errors) or by the previous stages of the system (segmentation and individual classification of the characters). In the experiment, the charac-

ters are classified by using a neural network, a perceptron with three layers trained by using the Back Propagation Training Algorithm [13]. This neural network has been trained by using 25 alphabets from 25 different authors. When an input character is obtained from the input text, the neural network produces a fuzzy symbol, $\tilde{y} = \{(x, \mu_{\tilde{y}}(x)) \mid x \in \Sigma\}$, where $\mu_{\tilde{y}}(x)$ is the neural output unit value associated with the character x of the alphabet Σ (26 letters).

The input texts are obtained from a large corpus of words known as *Brown Corpus* [8]. This corpus is formed by 15 different scope texts extracted from real life documents. We have taken a portion (about 6000 words) of each text. These texts are referred as *Brown-A*, *Brown-B*, ..., *Brown-R*. Taking the words of those texts, we have built-up the dictionaries *Dic-A*, *Dic-B*, ..., *Dic-R* (with an extension that ranges from 1268 to 2213 words).

3.1 Parameters of the system

The contextual post processing based on deformed system has been implemented by using the algorithm introduced in figure 1 modified to work with a set of words (dictionary), $D \subset \Sigma^*$. Several parameters of the algorithm must be fixed for the experiments like the fuzzy operations for computing transitions and the membership function values associated to transitions. The transitions have been computed using the Hamacher's t -conorm and t -norm [7].

We have selected the membership function values associated to transitions to be independent of symbols and states, so only three values μ_d , μ_i , μ_c are associated to delete, insert and substitute operation respectively.

In order to determine the values for the three edit operations and the γ parameter, we use a genetic algorithm [5]. The genetic algorithm can be formalized as a function, $GA(O, D, MD, \{\mu\}, \gamma)$, where O is the set of observed strings of fuzzy symbols; D the set of patterns (dictionary); MD the deformed fuzzy automaton; $\{\mu_d, \mu_i, \mu_c\}$ the membership values of the transitions and γ the parameter of the parametric t -conorm/ t -norm. The goal of the genetic algorithm is to determinate the parameter values that minimize the mean quadratic error of the system: $J(O) = (1/2) \sum_{\tilde{\alpha} \in O} ||target(\tilde{\alpha}) - out(\tilde{\alpha})||$. In that equation $out(\tilde{\alpha}) = [MD(\omega_1, \tilde{\alpha}), \dots, MD(\omega_n, \tilde{\alpha})]$. Each component represents the similarity of α with one of the patterns. As $\tilde{\alpha}$ is associated to a unique pattern in the training set O , the desired output is a vector $target(\tilde{\alpha}) = [a_1, \dots, a_i, \dots, a_n]$, such that $target(\tilde{\alpha})[i] = 1$ if and only if $\tilde{\alpha}$ is an element of the class ω_i ; in other case, $target(\tilde{\alpha})[i] = 0$.

In order to evaluate the error of the fuzzy automaton when classifying a given observed string $\tilde{\alpha}$, we are going to consider as metric the euclidean distance: $||target(\tilde{\alpha}) - out(\tilde{\alpha})||$. The selected optimization function will allow to select those parameters that maximize the similarity between the observed string of fuzzy symbols, and its associated pattern also maximize the difference between the observed string and the other patterns.

We have used a genetic algorithm with 10 populations, evolving during 300 evolution steps. At each step, the best four populations remain in the system for the next step of evolution. The other six are generated by selecting pairs of populations. The crossing and mutation probabilities are 0.25 and 0.1 respectively.

In order to assure that the *deformed system* does not increase the number of erroneous words, we have made experiments without introducing errors. In those tests, we obtained a 100% of recognition rate. We conclude that the *deformed system* does not create errors by itself.

	% Error Rate			
	31.3	44.7	71.6	87.7
% Recognition Rate	95.61	93.26	84.81	66.93
% Error Reduction Rate	86.0	84.9	78.8	62.4
Deviation (50 iterations)	0.723	0.702	0.575	0.550
% Class. Error Reduction Rate	95.4	94.6	90.7	74.7
% Insertion Error Reduction Rate	91.1	90.0	82.2	57.4
% Deletion Error Reduction Rate	74.1	73.5	67.9	51.4
% Substitution Error Reduction Rate	84.5	83.7	78.2	58.7
Poss (Recognition Rate 99%)	4	5	15	15

Figure 3: Results obtained for different error rate. Input text: *Brown-A*. Dictionary: *Dic-A* (1725 words).

3.2 Experimental results

In a first experiment, we introduce texts with different error rates in order to examine the capability of the *deformed system* to correct those errors. Figure 3 presents the results obtained in this experiment (it shows the average values obtained after 50 experiments). Besides the error reduction rate (with its standard deviation), the table also shows the percentages of recognition rate (correct words), corrected classification errors, corrected insertion errors, corrected deletion errors and corrected substitution errors. The row labelled 'Poss' indicates the interval (when the output of the *deformed system* is given in the rank level [17]) where, more than 99% of the corrected words fall into. For example, if Poss= 4 we assure that more than the 99% of corrected words are between the first and the fourth position when the output of the automaton is given in the rank level. Such a rank output becomes useful when a higher level of context (i.e., syntactic or semantic analysis) is wanted to be used.

	ER	RR	DRR	ERR	CLerr	Ierr	Derr	Cerr	Poss
Dic-B (2087 p.)	30.223	95.100	0.249	83.787	95.237	88.435	70.806	81.987	5
Dic-C (2213 p.)	30.503	95.181	0.241	84.203	94.623	89.455	72.071	81.996	4
Dic-D (1658 p.)	30.345	95.338	0.288	84.636	95.255	89.033	73.199	82.591	4
Dic-E (2063 p.)	34.285	96.149	0.245	88.768	96.295	91.601	80.532	88.114	4
Dic-F (1918 p.)	30.347	95.361	0.227	84.175	94.686	89.168	73.147	83.101	5
Dic-G (1899 p.)	31.195	95.726	0.261	86.300	96.015	90.564	74.835	84.940	4
Dic-H (1571 p.)	32.139	96.031	0.212	87.649	96.000	91.890	77.714	86.120	4
Dic-J (1480 p.)	31.706	95.969	0.237	87.285	96.448	90.039	77.622	86.152	4
Dic-K (1268 p.)	30.790	96.132	0.253	87.439	96.124	91.055	78.311	85.961	4
Dic-L (1449 p.)	25.376	94.627	0.281	78.828	92.865	84.465	63.290	75.383	5
Dic-M (1725 p.)	28.181	94.908	0.243	81.933	93.957	87.968	67.204	79.543	5
Dic-N (1447 p.)	25.880	94.454	0.246	78.568	92.942	85.939	61.122	74.938	5
Dic-P (1720 p.)	26.943	94.478	0.235	79.504	93.219	85.723	63.619	76.255	5
Dic-R (2078 p.)	28.717	95.089	0.265	82.900	94.173	87.971	69.918	80.386	5

Figure 4: Results (in percentage) obtained for different text domains. ER: Error Rate; RR: Recognition Rate; DRR: Standard Deviation of RR for 50 experiences; ERR: Error Reduction Rate; CLerr: Classification Error Reduction Rate; Ierr: Insertion Error Reduction Rate; Derr: Deletion Error Reduction Rate; Cerr: Substitution Error Reduction Rate; Poss: number of words needed for 99% recognition rate.

	ER = 31.3%		ER = 44.7%		ER = 71.6%	
	RR	ERR	RR	ERR	RR	ERR
Levenshtein distance	89.811	67.451	84.309	65.742	65.296	51.540
Max-prod (deformed)	93.014	77.516	90.049	77.650	80.597	72.833
Hamacher (deformed)	95.677	86.154	93.449	85.340	86.011	80.463

Figure 5: Influence of fuzzy symbols and the t -conorm/ t -norm in recognition rate. Input text: *Brown-A*. Dictionary: *Dic-A* (1725 words). ER: Error Rate; RR: Recognition Rate; ERR: Error Reduction Rate.

In a second experiment we have evaluated the influence of the text domain. Figure 4 shows the results obtained with the texts *Brown-B* (*Dic-B*) to *Brown-R* (*Dic-R*). Again, we introduce an error every 15 characters (*segment length*). This corresponds to an error rate from 25.4% to 34.3% depending on the text. We can see how the best correction rate is obtained with the *Brown-E* text. This is so because such a text has the lowest percentage (58.56%) of short words (less than six characters). Note that short words are difficult to correct because they have less context. On the other hand, the worst correction rate is obtained with the *Brown-N* text. This text has the highest percentage of short words (79.68%).

Finally, we made an experiment intended to evaluate the influence of fuzzy symbols and the fuzzy operations chosen for the transitions of the automata (see figure 5). In the first line the results correspond to the use of Levenshtein Distance [9] in order to compare the observed strings with the words in the dictionary. In this case the observed strings are composed of individual characters, being those that present greater value in the membership function in the corresponding fuzzy symbols. The edit operation costs used by the algorithm [15] implementing the Levenshtein Distance have unitary values. Therefore, an observed string will be classified as the word in the dictionary having the smallest number of edit operations.

As it was proved in [4], when the fuzzy automaton uses as t -conorm the *maximum* and as t -norm the *algebraic product*, it permits to calculate the Levenshtein Distance giving appropriate values to the transitions modelling the edit operations. In this way, giving values $\mu_i = 0.1$, $\mu_d = 0.1$, $\mu_c = 0.1$ one would obtain the same results as obtained by using the Levenshtein Distance. However, we can provide a deformed system for this fuzzy automaton to work with strings of fuzzy characters. Therefore, it is possible to evaluate the influence by taking into account all the information contained in the fuzzy symbols. These results are shown in the second line of figure 5. The third line shows the results obtained with the deformed system using Hamacher's t -conorm and t -norm together with the used values for the transitions in previous experiments. This enables us to compare the influence of the election of fuzzy operations.

In figure 5 the average values obtained through a simulation with one hundred texts are shown. The best results are obtained with the deformed system method. Differences between Levenshtein, max-prod and Hamacher are minor if the number of errors per word is low (error rate 31.3%), and major if the number of errors per word is high (error rate 71.6%). Differences between max-prod and Hamacher are relevant in terms of error reduction rate (near eight points). When a deformed system is used with a high number of errors per word, RR and ERR become higher.

4 CONCLUSIONS

In this work we have introduced a fuzzy method that allows recognizing imperfect strings of fuzzy symbols. The basic idea is to use a deformed system built from a fuzzy automaton. The fuzzy automaton is defined in such a way that its transitions model every possible edit operation. Thus, it allows to compute the similarity between an observed string and a pattern string. When this fuzzy automaton is extended by means of a deformed system, the same computation can be performed over strings of fuzzy symbols.

Our method is quite general. It does not assume a limit in the number of errors to be handled. Moreover, the deformed system allows to adjust the values of transition fuzziness and to select different fuzzy operations for computing the transitions. We wish to emphasize the influence that the fuzzy operations used when computing the transitions has on the obtained results. We have used the parametric t -norm/ t -conorm proposed by Hamacher [7]. Variations of the t -norm/ t -conorm parameter lead to very different results.

Due to the difficulty to adjust the values of transition fuzziness once selected the fuzzy operators, we have used a genetic algorithm that obtains good results.

The deformed system has been used as the layer of contextual post processing in a system of text recognition. Section 3 shows that the deformed system gives high ratios of error recovery even when the number of edition errors considered is high.

It has been clearly established that, although usually the string matching methods are based on discrete mathematics and probabilistic techniques, fuzzy techniques can also be useful within this area.

Acknowledgments

The authors wish to thank to the enterprise Investigación y Programas, S.A. (IPSA) for the contribution to the experimental part and for the financial support through research grant OTRI-2001,4059 (Universidad Pública de Navarra).

References

- [1] H. Bunke, J. Csirik, Parametric String Edit Distance and its Application to Pattern Recognition, *IEEE Transactions on Systems, Man and Cybernetics* **25**, No. 1, (1995) 202–206
- [2] J. Echanobe, J.R. González de Mendivil, J.R. Garitagoitia, C.F. Alastruey, Deformed systems for contextual postprocessing, *Fuzzy Sets and Systems* **96**, (1998) 335–341
- [3] P. Gader, M. Mohamed, J.H. Chiang, Comparison of Crisp and Fuzzy Character Neural Networks in Handwritten Word Recognition, *IEEE Transactions on Fuzzy Systems* **3**, No. 3, (1995) 357–510
- [4] J.R. González de Mendivil, J.R. Garitagoitia, J.J. Astrain, Fuzzy automata for imperfect string matching, *Proceedings of ESTYLF 2000*, Sevilla, Spain, (2000) 527–532
- [5] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company, Reading, Massachusetts, USA (1989)
- [6] J.J. Hull, S.N. Srihari, R. Choudhari, An integrated Algorithm for Text Recognition: Comparison with a Cascaded Algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **5**, No. 4, (1983) 384–395
- [7] G.J. Klir, B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice Hall, New Jersey, USA (1995)

- [8] H. Kucera, W.N. Francis, *Computational Analysis of Present-Day American English*, RI Brown Univ. Press, Providence, USA (1967)
- [9] V.I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals, *Soviet Physics Doklady*, **10**, No. 8, (1966) 707–710
- [10] A. Marzal, E. Vidal, Computation of Normalized Edit Distance and Applications, *IEEE PAMI* **15**, No. 9, (1993) 926–932
- [11] C.V. Negoita, D.A. Ralescu, *Application of Fuzzy Sets to System Analysis*. Birkaeuser, Basilea, Switzerland (1975)
- [12] B.J. Oommen, R.L. Kashyap, A formal theory for optimal and information theoretic syntactic pattern recognition, *Pattern Recognition* **31**, No. 8, (1998) 1157–1177
- [13] D.E. Rumelhart, G.e. Hinton, R.J. Williams, Learning Internal Representations by Error Propagation In: Rumelhart & McClelland Eds (eds): *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, **1**, Foundations. MIT Press (1986)
- [14] D. Sankoff, J.B. Kruskal, *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*, Addison-Wesley Publishing Company, Reading, Massachusetts, USA (1983)
- [15] R.A. Wagner, M.J. Fischer, The string-to-string correction problem In: *Journal of ACM* **21**, No. 1, (1974) 168–173
- [16] Wen-Tsong Chen, P. Gader, Hongchi Shi, Lexicon-Driven Handwritten Word Recognition Using Optimal Linear Combinations of Order Statistics, *IEEE PAMI* **21**, No. 1, (1999) 77–82
- [17] L. Xu, A. Krzyzak, C.Y. Suen, Methods for Combining Multiple Classifiers and Their Applications to Handwriting Recognition, *IEEE Trans. Sys. Man. and Cyber. (SMC)* **22**, No. 3, (1992) 418–435

An XML-based specification of fuzzy logic controllers

Dario Mastropasqua, Nicola Mosca, Fabio Zambetta
VALIS Group
Dipartimento di Informatica, Università degli Studi,
Via E. Orabona 4, Bari, I-70124

Abstract. Since their introduction, Fuzzy Systems have proven their usefulness in various fields, including linear and non linear control, pattern recognition, financial systems and data analysis.

Current fuzzy systems, however, vary in supported capabilities, rules representation and storage. These differences, strictly interconnected, cause a lot of problems when there is a necessity to port a whole set of fuzzy rules from one system to another. To solve this portability issue, the International Electrotechnical Commission (IEC) formed a specific committee to propose a standard format to represent fuzzy systems, named Fuzzy Control Language (FCL).

Subsequently, in another context, another research group have proposed a standardized system to represent Evolutionary Computation Systems using an XML-based grammar. Based on this idea, strengthened by growing Internet fame, we propose a mapping of the FCL grammar and capabilities in a standardized XML format, showing at the same time, a bunch of extensions that can further enhance FCL language expressive power.

1. Introduction

Nowadays, there is a great demand for systems capable to share data among them and optionally, capable of cooperation abilities. To satisfy these demands, a large number of standards have been proposed and adopted. SGML derived languages, such as XML (<http://www.w3.org/XML>), HTML (<http://www.w3.org/MarkUp>), and VRML (<http://www.web3d.org>), have acquired a large popularity (partially) solving representation issues for various structural discriminate environments. Meanwhile another bunch of useful standards have been proposed to publish services remotely through Simple Object Access Protocol (SOAP). SOAP (<http://www.w3.org/2000/xp/Group>) uses an XML-based grammar to describe functions that can be used through various programming languages, such as C++, Java, and even Visual Basic. EAML [1] is another step toward the development of sharable and distributed systems.

The fuzzy system community, in another way, have had necessity to exchange portable Fuzzy Control programs among different programming systems.

There were some attempts to define fuzzy systems using XML for specific contexts, such as in the work being done by Teuteberg (http://www.wiwi.euv-frankfurt-o.de/wiwi/en/team/members/fteuteberg/PDF_Dateien/fuzzyl.pdf), in the application of fuzzy classification for personnel acquisition. Scientio proposes a commercial system, XML Rule [2], to help define fuzzy rules in traditional *if-then* style using the English language, converting automatically them to XML format using Javascript and XSL transforms.

Turowski and Weng [3] introduce a formal syntax for important fuzzy data types used to store fuzzy information, accompanied by its DTD and specific Java tools provided to aid fuzzy system development. None of these systems attempts to preserve previously introduced proposal of standard, such as FCL [4].

To further expand the choice of Soft Computing systems qualified to operate in distributed environments, we propose a mapping of the FCL [4] grammar and capabilities in a standardized XML format, showing at the same time, a bunch of extensions that can further enhance FCL language expressive power. Our proposed fuzzy language grammar definition, named Fuzzy Applications Markup Language (FAML), combines FCL with the advantages of XML-based standards, such as extensibility and multi-platform portability. XML hierarchical structure is also well suited to represent systems which can be described with elements arranged in a nested order.

2. XML advantages

XML emerged as a way to overcome some defects of its two predecessors, Standard Generalized Markup Language (SGML) and HyperText Markup Language (HTML). SGML, the international standard for marking up data, has been used since the 80's and is an extremely powerful and extensible tool for semantic markup which is particularly useful for cataloging and indexing data. However, SGML is somewhat complex. HTML was originally designed at CERN around 1990 to provide a very simple version of SGML which could be easily used. Unfortunately, HTML is very limited in data cataloging.

In 1996, the World Wide Web Consortium (W3C) decided to sponsor a group of SGML experts to define a subset of SGML, named XML. Finally, in 1998, the W3C approved Version 1.0 of the XML specification.

XML stands for eXtensible Markup Language and therefore is a document and data description meta-markup language that provides a flexible way to create common information formats and shares both the format and the data on the World Wide Web, intranets, and elsewhere. Its support for Unicode character set provides a useful way to use XML in multi-lingual contexts. XML can be used with legacy systems and data and to support loose coupling between different systems. It is worth noting that XML describes just the data. Data in XML format can then be presented in various ways through XSL translation (<http://www.w3.org/Style/XSL>). XSL is another W3C standard that specify how to format XML data. In this way, it is possible to decouple form from content, and through the use of different XSL stylesheets, render the data in very different ways. XML is hierarchical in nature and, as its name says, extensible in many ways such as extensible tag library, extensible document structure, and extensible document elements.

3. EAML

Veenhuis and others [1], in order to overcome lacks of standards in Evolutionary Algorithms (EA) specification, especially in the modeling, description and documentation of EA at once, have proposed EAML.

EAML is derived from XML and allows the specification of an EA in XML notation. Therefore such an XML document describes the EA on an abstract level, which supports the rapid design of a huge variety of evolutionary algorithms.

Moreover, a designed evolutionary-algorithm-model can also be used as compiler input to generate ready to use object-oriented code.

The basic concept of the model is founded in an element hierarchy, where operators and basic algorithms are handled as parameterized elements of a structural tree and where the structural tree describes the computational flow.

This approach enables a more abstract handling of the evolutionary algorithms, and how it speeds up the algorithm design.

4. FCL Basis

Lotfi Zadeh introduced the world to fuzzy logic in 1965 as a mathematical discipline that would allow designers to create systems directly from human intuition and experience [5]. A very large number of publications on fuzzy set theory exists, proposing methods, algorithms, and terminology on how to use fuzzy logic. The lack of standardization has resulted in many inconsistencies among these publications. The same terminology frequently refers to different methods, and identical algorithms are often referred to by different names [6].

Many groups in different countries have worked on the standardization of fuzzy logic. One of the standardization activities has already resulted in an extension of the existing International Electrotechnical Commission (IEC 1131) standard. This standard defines devices used for industrial automation from mechanical parameters to programming and documentation. The new section of this standard, IEC 1131-7, defines fuzzy logic components for industrial automation and process control, especially performed by Programmable Controllers.

FCL is part of IEC 1131-7 standard offering to the manufacturers and to the users a well defined common understanding of the basic means to integrate Fuzzy Control applications in the Programmable Controller languages, and the possibility to exchange portable Fuzzy Control programs among different programming systems.

Fuzzy Control is used from small and simple applications up to highly sophisticated and complex projects. To cover all kinds of usage, FCL specifies a few conformance levels. All FCL systems must comply to Basic Level, that supports declaration of input/output variables with membership functions with at most three points for input variables and degree of membership coordinate limited to either 0 or 1, only singletons for output variables, a unique defuzzification method suitable for singletons (COGS), and only one conclusion for each rule.

To further extend FCL usage, IEC 1131-7 specifies an Extension Level with various enhancements including membership functions of input and output variables defined with up to four points, a few more composition rules than *max-min* provided in Basic Level, and other defuzzification methods that are *center of gravity*, *center of area*, *left-most maximum* and *right-most maximum*. Extension Level, moreover, includes default values, a more complex variable-based weighting factor capability and allows to specify more than one output variable.

IEC to further expand these two levels, favors definition of Open Level elements, but it comes at the cost that FCL rules using Open Level elements, and even Extension Level elements, cannot be completely portable among systems compliant with IEC 1131-7 Basic Level.

5. FAML structure

FAML element organization strongly resembles standard XML structure and FCL structure both at once. FAML inherits element grammar specification of standard XML-derived languages with the nested order element declaration used in FCL. A fuzzy system is therefore described through a set of input and output fuzzy variables followed by a set of rules in a *condition*→*conclusion* style format.

FAML inherits some characteristics even from EAML. Although the latter covers a somewhat different area, there are useful elements in EAML that can be further exploited and included in FAML. One of these ideas is the inclusion of custom functions in XML sections. This can be used, for example, to implement user-defined defuzzification methods.

Here we report an example taken from IEC 1131-7 draft [4], for a gas control system, which is easy to describe through FCL. The system is aimed to stabilize temperature and pressure of gas into a container to certain levels, modifying container's valve aperture level. This valve can be inlet, closed, or drainage. Naturally valve, temperature and pressure can be represented in fuzzy terms. The interaction between gas container and the fuzzy controller is represented in the block diagram in Figure 1.

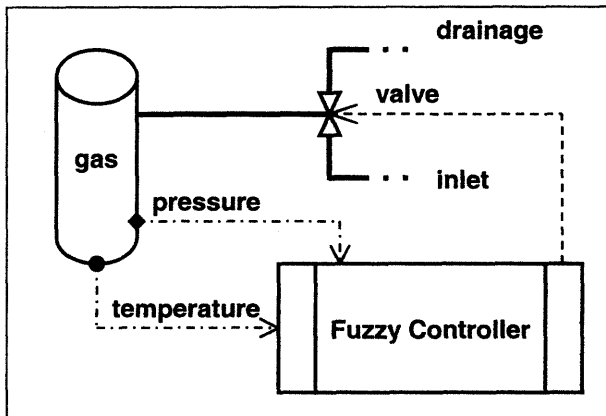


Figure 1. Block diagram of gas control system

To specify the fuzzy controller, we propose two listings, one using FCL syntax, and the latter using FAML syntax, in order to show the mapping between these two formats. FCL listing is reported in Figure 2.

```

FUNCTION_BLOCK Fuzzy_FB
VAR_INPUT
    temp:      REAL;
    pressure:  REAL;
END_VAR
VAR_OUTPUT
    valve:     REAL;
END_VAR
FUZZIFY temp
    TERM cold := (3,1) (27,0);
    TERM hot  := (3,0) (27,1);
END_FUZZIFY
  
```

```

FUZZIFY pressure
  TERM low := (55,1) (95,0);
  TERM high := (55,0) (95,1);
END_FUZZIFY
DEFUZZIFY valve
  TERM drainage := -100;
  TERM closed := 0;
  TERM inlet := 100;
  ACCU : MAX;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY
RULEBLOCK No1
  AND : MIN;
  RULE 1 : IF temp IS cold AND pressure IS low THEN valve IS inlet;
  RULE 2 : IF temp IS cold AND pressure IS high THEN valve IS closed
WITH 0.8;
  RULE 3 : IF temp IS hot AND pressure IS low THEN valve IS closed;
  RULE 4 : IF temp IS hot AND pressure IS high THEN valve IS drainage;
END_RULEBLOCK
END_FUNCTION_BLOCK

```

Figure 2. Gas control system description in FCL

The same example is reported in Figure 3 using FAML syntax. This syntax is longer than FCL equivalent due to XML inheritance.

```

<FLC type="Mamdani" name="Fuzzy_FB">
  <FUZZYVARS>
    <FUZZYVAR name="temp">
      <MEMBERSHIP type="Triangle" name="cold">
        <POINTS>3 3 27</POINTS>
      </MEMBERSHIP>
      <MEMBERSHIP type="Triangle" name="hot">
        <POINTS>3 27 27</POINTS>
      </MEMBERSHIP>
    </FUZZYVAR>
    <FUZZYVAR name="pressure">
      <MEMBERSHIP type="Triangle" name="low">
        <POINTS>55 55 95</POINTS>
      </MEMBERSHIP>
      <MEMBERSHIP type="Triangle" name="high">
        <POINTS>55 95 95</POINTS>
      </MEMBERSHIP>
    </FUZZYVAR>
    <FUZZYVAR name="valve">
      <MEMBERSHIP type="Singleton" name="drainage" >
        <POINTS>-100</POINTS>
      </MEMBERSHIP>
      <MEMBERSHIP type="Singleton" name="closed" >
        <POINTS>0</POINTS>
      </MEMBERSHIP>
      <MEMBERSHIP type="Singleton" name="inlet" >
        <POINTS>100</POINTS>
      </MEMBERSHIP>
    </FUZZYVAR>
  </FUZZYVARS>
  <INVARS>
    <IDS>
      <ID name="temp"/>
      <ID name="pressure"/>
    </IDS>
  </INVARS>

```

```

<OUTVARS>
  <IDS>
    <ID name="valve" defuzzify="COGS" default="0" accum="MAX"/>
  </IDS>
</OUTVARS>
<RULEBASE>
  <RULES name="No1" composition="MIN">
    <RULE>
      <DATAS>
        <LHS_DATA>
          <DATA var="temp" set="hot"/>
          <DATA var="pressure" set="low"/>
        </LHS_DATA>
        <RHS_DATA>
          <DATA var="valve" set="inlet"/>
        </RHS_DATA>
      </DATAS>
    </RULE>
    <RULE>
      <DATAS>
        <LHS_DATA>
          <DATA var="temp" set="cold"/>
          <DATA var="pressure" set="high"/>
        </LHS_DATA>
        <RHS_DATA>
          <DATA var="valve" set="closed" weight="0.8"/>
        </RHS_DATA>
      </DATAS>
    </RULE>
    <RULE>
      <DATAS>
        <LHS_DATA>
          <DATA var="temp" set="hot"/>
          <DATA var="pressure" set="low"/>
        </LHS_DATA>
        <RHS_DATA>
          <DATA var="valve" set="closed" />
        </RHS_DATA>
      </DATAS>
    </RULE>
    <RULE>
      <DATAS>
        <LHS_DATA>
          <DATA var="temp" set="hot"/>
          <DATA var="pressure" set="high"/>
        </LHS_DATA>
        <RHS_DATA>
          <DATA var="valve" set="drainage" />
        </RHS_DATA>
      </DATAS>
    </RULE>
  </RULES>
</RULEBASE>
</FLC>

```

Figure 3. Gas control system description in FAML

FAML and FCL descriptions are pretty similar, except for some details. One difference is due to the possibility in FAML to use the same variable as input and output.

To achieve this, FAML structure consists logically of three parts:

1. Fuzzy variables definition
2. Input and output system variables specification
3. Rules specification

Using this capability, it is possible to maintain states between successive evaluations in the form of input and output variables that share the same name, and therefore, are implicitly connected each others.

Another major difference is given by grammar details of FAML rules specification, where FCL English-style *if-then* pairs are specified in a more structured way through a hierarchical distinction between left handed side data (LHS data) which describe the condition part and right handed side data (RHS data) which describe the conclusion part. This difference, that at first may be considered as an ugly idea that makes FAML rules far less readable than FCL rules, can be efficiently handled through XSL translation. This choice makes instead possible to parse rules with standard XML management strategies avoiding user defined syntax checking.

Naturally, there are a number of disadvantages to this choice, including the prolific size associated with XML encoding, and the loss of human readability in absence of suitable tools. Former disadvantage does not limit the usability of the method, and is shared with all XML-based standards. Therefore, in our point of view, W3C will address and solve the problem very soon. Latter disadvantage can be solved, as suggested, through suitable tools on which we will shortly focus lately on this document.

It is worth noting another major difference between the two structures. FCL fuzzy sets can be defined with maximum three points membership functions for the FCL-“basic” version, which must be supported by all FCL-compliant systems, and up to four points for the “extended” version. On the other side, in FAML we propose to support membership functions defined with more than four points.

FCL limited capabilities to deal with complex membership functions is probably due to its focus to embedded systems and programmable controllers. FAML, indeed, primarily addresses generic personal computer systems, with a particular focus on distributed systems and web-based applications.

Moreover FAML proposal comes four years after initial IEC 1131-7 first draft, and current computational power makes possible to use even fuzzy sets based on Bell and Gaussian curves in real-time applications.

As in FCL, FAML does not require that membership functions of some fuzzy variable must be of the same type; in any case, FAML requires an explicit declaration of the membership function described, in order to optimize successive fuzzy evaluations.

FAML does allow to use standard membership function types such as singleton, triangular, rectangular and trapezoidal. It is also possible to define “composite” memberships. These allow specifying membership functions with more than three points, virtually without no-upper bound limits. Composite type membership functions can be used to sample Bell or Gaussian curves, or to define other non standard membership functions. Another benefit of composite fuzzy set is the ability to specify degree of membership coordinate in the range (0, 1).

FAML will be able to support even countinuous membership functions like the already cited Bell and Gaussian curves as well as Sigmoidal, Exponential and others, where it is required more precision and performance is not a problem.

Rules in FAML, unlike FCL, are all in conjunctive form and FAML does not allow to specify rules with OR clauses. This is not restrictive, because every rule using one or more disjunctions can be rewritten as a series of rules that does not make use of OR clauses.

As a final note, FAML as FCL can support subconditions that are in negative form. To show how to achieve this, we compare FCL syntax with FAML syntax in Table 1.

Table 1. Mapping between various negative forms

FCL	FAML
temp IS NOT hot	<DATA var="temp" set="hot" invert="true">
NOT (temp IS hot)	<DATA var="temp" set="hot" exclude="true">

6. Conclusions and Future Works

FAML file format, in our opinion, introduces various enhancements to FCL. There are lot of capabilities that can be used to design more complex fuzzy systems, including support for composite membership functions, sigmoidal and S-curve, support for variables that can be used simultaneously as input and output, and so on. FAML inherits also other advantages from XML that can be used in various ways.

We could imagine, for example, to provide various XSL sheets to display data in a multitude of modes, as well as show (or hide) only the particular elements we are interested in. As any XML-based standard, FAML format can be used flawlessly with many languages such as C++, Java, or Visual Basic.

We can even imagine to create servers exposing SOAP-based services on the Web that can use fuzzy systems specified in FAML. And naturally, we can share FAML systems with everyone, everywhere.

In any case, FAML can be just another proposal of standard if not supported by suitable tools. We will therefore focus on the creation of tools that make easy to edit fuzzy systems specified in FAML files, through our editor: FuzzyEd.

FuzzyEd, currently in development, will support the definition of fuzzy sets using the various membership functions described in this document, as well as a spreadsheet-like rule editor, implemented to rapidly develop fuzzy systems.

Another tool that will be developed is a converter between FAML files and FCL files. This utility will take care of converting files accounting the differences in structure and capabilities of the two formats. FAML specific features will not be available in FCL files, and conversion process will not be too smooth going from FAML to FCL, but it will make possible to port systems created in FCL to FAML based systems.

References

- [1] C. Veenhuis, K. Franke, M. Köppen, A Semantic Model for Evolutionary Computation, in Proc. 6th International Conference on Soft Computing (IIZUKA). Iizuka, 2000
- [2] Edmonds, XML Rule Batch White Paper, Scientio, 2001
- [3] K. Turowski, U. Weng, Representing and processing fuzzy information – an XML-based approach, in Elsevier, Knowledge Based Systems 15, 2002
- [4] IEC 1131, Part 7 – Fuzzy Control Programming, Committee Draft CD 1.0
- [5] Zadeh, L.A.: *Fuzzy Sets*. Information and Control, 1965.
- [6] Zimmermann, H.-J., *Fuzzy Set Theory - and its applications*, Second Revised Edition (1991), Boston, Dordrecht, London.

Comparison of Fuzzy Rule Selection Criteria for Classification Problems

Hisao ISHIBUCHI and Takashi YAMAMOTO

*Department of Industrial Engineering, Osaka Prefecture University
 Sakai, Osaka 599-8531, Japan
 {hisaoi, yama}@ie.osakafu-u.ac.jp*

Abstract. This paper compares heuristic rule selection criteria in fuzzy rule extraction for classification problems. Using several heuristic criteria, we examine the performance of extracted fuzzy rules through computer simulations on four data sets (glass, Wisconsin breast cancer, wine, and sonar). Simulation results show that better results are obtained from composite criteria of the confidence and support measures than the individual use of those measures. It is also suggested that genetic algorithm-based rule selection can improve the classification ability of extracted fuzzy rules by searching for good rule combinations. This result shows the importance of taking into account the combinatorial effect (i.e., interaction) of extracted fuzzy rules when we design fuzzy rule-based systems.

1. Introduction

In the design of fuzzy rule-based systems, there exist two conflicting objectives: error minimization and comprehensibility maximization. The error minimization has been used in many applications (e.g., fuzzy control, fuzzy modeling, and fuzzy classification). While the comprehensibility was not usually taken into account in those applications, recently the tradeoff between these two objectives has been discussed in some studies (e.g., see [1]).

When fuzzy rule-based systems are used for two-dimensional problems, fuzzy rules can be represented in a tabular form. Figure 1 shows an example of a fuzzy rule table for a two-dimensional pattern classification problem. In this figure, we have four fuzzy rules:

- If x_1 is *small* and x_2 is *small* then Class 1, (1)
- If x_1 is *small* and x_2 is *large* then Class 2, (2)
- If x_1 is *large* and x_2 is *small* then Class 3, (3)
- If x_1 is *large* and x_2 is *large* then Class 4, (4)

where *small* and *large* are linguistic values defined by triangular membership functions. As shown in Figure 1, fuzzy rules for two-dimensional problems can be written in a human understandable manner using the tabular form representation. When fuzzy rule-based systems are applied to high-dimensional problems, their comprehensibility is significantly degraded due to the two difficulties: the increase in the number of fuzzy rules and the increase in the number of antecedent conditions of each fuzzy rule.

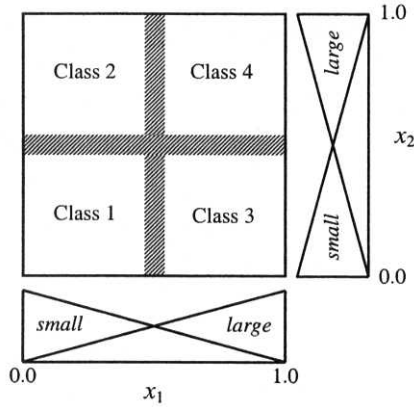


Figure 1. Four fuzzy rules in the two-dimensional pattern space $[0, 1] \times [0, 1]$.

In the field of knowledge discovery and data mining [2], emphasis is placed on the comprehensibility of extracted rules. Fuzzy rule-based systems have an inherent advantage with respect to their comprehensibility over other nonlinear systems (e.g., neural networks). This is because fuzzy rules are linguistically interpretable. Such an inherent advantage, however, is significantly degraded due to the above-mentioned two difficulties when fuzzy rule-based systems are applied to high-dimensional problems. For finding comprehensible fuzzy rule-based systems for high-dimensional classification problems, fuzzy rule extraction was formulated as a three-objective optimization problem in [3] where the classification performance was maximized, the number of fuzzy rules was minimized, and the number of antecedent conditions was minimized. Three-objective GBML (genetics-based machine learning) algorithms were used for finding non-dominated rule sets with respect to the three objectives. Because the number of possible fuzzy rules exponentially increases with the number of attributes (i.e., with the dimensionality of problems), the genetic search needs long CPU time in the case of high-dimensional problems. Thus iterative fuzzy GBML algorithms have been proposed for extracting fuzzy rules for high-dimensional problems based on heuristic rule selection criteria (e.g., [4]-[6]). In such an iterative algorithm, a single fuzzy rule is generated from its single iteration.

The aim of this paper is to compare several heuristic rule selection criteria used for the fuzzy rule extraction from numerical data. In our computer simulations, we extract a pre-specified number of fuzzy rules using such a heuristic criterion. The performance of extracted fuzzy rules is examined on some data sets with many continuous attributes available from the UCI ML repository. Simulation results show that better results are obtained from composite criteria of the confidence and support measures than their individual use. Simulation results also show that any heuristic criteria do not always generate fuzzy rules with high classification performance when we use a naive greedy method for rule extraction (i.e., when we simply extract a pre-specified number of the best fuzzy rules with respect to a heuristic criterion without taking into account the combinatorial effect of extracted fuzzy rules). Finally we show that genetic algorithm-based rule selection can improve the classification ability of extracted fuzzy rules. This means that heuristic rule selection criteria can be used as a pre-screening tool of candidate fuzzy rules in genetic algorithm-based rule selection [7], [8].

2. Fuzzy Rules for Classification Problems

For classification problems with n attributes, we use fuzzy rules of the following form:

$$\text{Rule } R_q: \text{If } x_1 \text{ is } A_{q1} \text{ and } \dots \text{ and } x_n \text{ is } A_{qn} \text{ then Class } C_q \text{ with } CF_q, \quad (5)$$

where R_q is the label of the q -th rule, $\mathbf{x} = (x_1, \dots, x_n)$ is an n -dimensional pattern vector, A_{qi} is an antecedent fuzzy set (i.e., linguistic value such as *small* and *large* in Figure 1), C_q is a class label, and CF_q is a rule weight [9].

First we explain how the consequent class C_q and the rule weight CF_q of the fuzzy rule R_q in (5) are specified from numerical data. Let us assume that we have m labeled patterns $\mathbf{x}_p = (x_{p1}, \dots, x_{pn})$, $p = 1, 2, \dots, m$ from M classes (i.e., we have an n -dimensional M -class problem). We define the compatibility grade of each training pattern \mathbf{x}_p with the antecedent part $\mathbf{A}_q = (A_{q1}, A_{q2}, \dots, A_{qn})$ of the fuzzy rule R_q using the product operator as

$$\mu_{\mathbf{A}_q}(\mathbf{x}_p) = \mu_{A_{q1}}(x_{p1}) \cdot \mu_{A_{q2}}(x_{p2}) \cdot \dots \cdot \mu_{A_{qn}}(x_{pn}), \quad (6)$$

where $\mu_{A_{qi}}(\cdot)$ is the membership function of the antecedent fuzzy set A_{qi} . The fuzzy conditional probability $\text{Pr}(\text{Class } h | \mathbf{A}_q)$ of Class h ($h = 1, 2, \dots, M$) for the antecedent part \mathbf{A}_q is numerically approximated as follows [10]:

$$\text{Pr}(\text{Class } h | \mathbf{A}_q) \equiv \frac{\sum_{\mathbf{x}_p \in \text{Class } h} \mu_{\mathbf{A}_q}(\mathbf{x}_p)}{\sum_{p=1}^m \mu_{\mathbf{A}_q}(\mathbf{x}_p)}. \quad (7)$$

The right-hand side of (7) is often referred to as the confidence of the fuzzy association rule " $\mathbf{A}_q \Rightarrow \text{Class } h$ " in the field of fuzzy data mining [11], [12]. This definition of the confidence is a natural extension of its non-fuzzy version [13], [14]. That is, the confidence of the fuzzy association rule " $\mathbf{A}_q \Rightarrow \text{Class } h$ " is defined as follows [11], [12].

$$c(\mathbf{A}_q \Rightarrow \text{Class } h) = \frac{\sum_{\mathbf{x}_p \in \text{Class } h} \mu_{\mathbf{A}_q}(\mathbf{x}_p)}{\sum_{p=1}^m \mu_{\mathbf{A}_q}(\mathbf{x}_p)}. \quad (8)$$

The consequent class C_q of the fuzzy rule R_q is specified by identifying the class with the maximum fuzzy conditional probability (i.e., the maximum confidence). That is, we specify the consequent class C_q as follows:

$$c(\mathbf{A}_q \Rightarrow \text{Class } C_q) = \max \{c(\mathbf{A}_q \Rightarrow \text{Class } h) | h = 1, 2, \dots, M\}. \quad (9)$$

On the other hand, there exist several alternative methods for specifying the rule weight CF_q [15]. The choice of an appropriate specification depends on a fuzzy reasoning method used for pattern classification [10]. The specification of the rule weight of each fuzzy rule has a large effect on the classification performance of fuzzy rule-based systems [16].

In this paper, we use a single winner-based method [17]. Let S be the set of fuzzy rules in our fuzzy rule-based system. A single winner rule R_w is chosen from the rule set S for an input pattern \mathbf{x}_p as

$$\mu_{\mathbf{A}_w}(\mathbf{x}_p) \cdot CF_w = \max \{\mu_{\mathbf{A}_q}(\mathbf{x}_p) \cdot CF_q | R_q \in S\}. \quad (10)$$

That is, the winner rule has the maximum product of the compatibility grade and the rule weight in the fuzzy rule-based system. See [10], [17], [18] for other fuzzy reasoning methods for pattern classification.

When we use the single winner-based method, the following definition of the rule weight CF_q is appropriate for two-class problems [10], [15]:

$$CF_q = \begin{cases} c(\mathbf{A}_q \Rightarrow \text{Class 1}) - c(\mathbf{A}_q \Rightarrow \text{Class 2}), & \text{if } C_q = \text{Class 1}, \\ c(\mathbf{A}_q \Rightarrow \text{Class 2}) - c(\mathbf{A}_q \Rightarrow \text{Class 1}), & \text{if } C_q = \text{Class 2}. \end{cases} \quad (11)$$

The point is the extension of this formulation to the case of multi-class problems. In this paper, we use the following definition (which is the fourth definition in [15]) because good results were obtained from this definition in our preliminary computer simulations.

$$CF_q = c(\mathbf{A}_q \Rightarrow \text{Class } C_q) - c(\mathbf{A}_q \Rightarrow \text{Class } \overline{C_q}), \quad (12)$$

where

$$c(\mathbf{A}_q \Rightarrow \text{Class } \overline{C_q}) = \sum_{\substack{h=1 \\ h \neq C_q}}^M c(\mathbf{A}_q \Rightarrow \text{Class } h). \quad (13)$$

In this definition, our M -class pattern classification problem is virtually handled as a two-class problem where classification is performed between Class C_q and a merged class including all the other classes (i.e., $h = 1, 2, \dots, M$; $h \neq C_q$).

As shown in [16], fuzzy rule-based systems can generate various classification boundaries by adjusting the rule weight of each rule even when we use fixed membership functions. In Figure 2, we show some examples of classification boundaries generated by the four fuzzy rules in Figure 1 using different rule weights. It should be noted that the membership function of each linguistic value in Figure 1 is not modified in Figure 2. A real number in each decision area in Figure 2 shows the rule weight of the corresponding fuzzy rule. As we can see from this figure, classification boundaries are not always parallel to the axes of the pattern space. This is a characteristic feature of fuzzy rule-based classification.

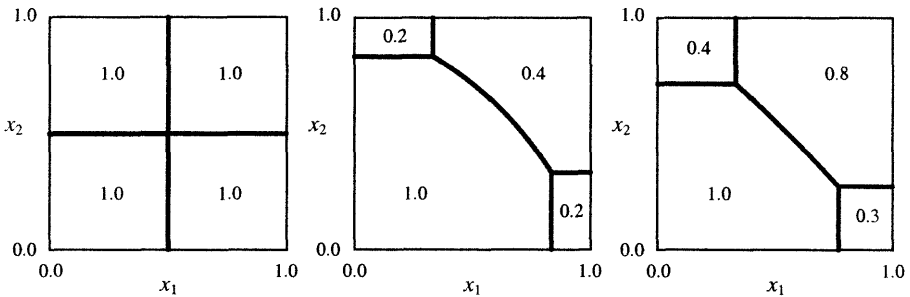


Figure 2. Some examples of classification boundaries generated by the four fuzzy rules in Figure 1.

3. Rule Selection Criteria

In the field of data mining, the confidence and the support of association rules have been often used as rule selection criteria [13], [14]. We have already shown an extension of the confidence to the case of fuzzy rules in (8). In the same manner, the support is defined for the fuzzy association rule " $\mathbf{A}_q \Rightarrow \text{Class } h$ " as follows [11], [12]:

$$s(\mathbf{A}_q \Rightarrow \text{Class } h) = \frac{1}{m} \sum_{\mathbf{x}_p \in \text{Class } h} \mu_{\mathbf{A}_q}(\mathbf{x}_p), \quad (14)$$

where m is the number of given training patterns. In our former studies [12], we used the confidence, the support and their product as rule selection criteria for extracting a pre-specified number of fuzzy rules from numerical data. Simulation results in [12] showed that the product criterion of the confidence and the support outperformed their individual use. In this paper, we examine two composite criteria of the confidence and the support in addition to their product. One is the support criterion with the minimum confidence level where the rule selection is performed using the support criterion from fuzzy rules whose confidence values are larger than or equal to a pre-specified minimum level. The other is the confidence criterion with the minimum support level.

In an iterative fuzzy GBML algorithm called SLAVE [4], [5], a single fuzzy rule was extracted from numerical data in its single iteration using a heuristic rule selection criterion. While a somewhat complicated general formulation was shown in [4], [5], the rule selection criterion in their computer simulations was very simple: $n^+(R) - n^-(R)$ where $n^+(R)$ and $n^-(R)$ are the number of positive and negative examples, respectively. This measure can be fuzzified using the compatibility grade of each pattern in (6) as

$$\begin{aligned} f_{\text{SLAVE}}(R_q) &= n^+(R_q) - n^-(R_q) \\ &= \sum_{\mathbf{x}_p \in \text{Class } C_q} \mu_{A_q}(\mathbf{x}_p) - \sum_{\mathbf{x}_p \notin \text{Class } C_q} \mu_{A_q}(\mathbf{x}_p). \end{aligned} \quad (15)$$

This formulation can be equivalently rewritten by dividing the right-hand side by m as

$$f_{\text{SLAVE}}(R_q) = s(A_q \Rightarrow \text{Class } C_q) - s(A_q \Rightarrow \text{Class } \overline{C_q}), \quad (16)$$

where

$$s(A_q \Rightarrow \text{Class } \overline{C_q}) = \sum_{\substack{h=1 \\ h \neq C_q}}^M s(A_q \Rightarrow \text{Class } h). \quad (17)$$

On the other hand, the following measure is used in an iterative fuzzy GBML algorithm for the learning of maximal structure fuzzy rules in Castro et al. [6]:

$$f_{\text{Castro}}(R_q) = \frac{n^+(R_q)}{|\text{Class } C_q|} \times \frac{|\text{Class } \overline{C_q}| - n^-(R_q)}{|\text{Class } \overline{C_q}|}, \quad (18)$$

where $|\text{Class } C_q|$ and $|\text{Class } \overline{C_q}|$ are the number of training patterns in Class C_q and the other classes, respectively. This formulation can be fuzzified using the compatibility grade of each pattern in (6) and equivalently rewritten by dividing the right-hand side by m^2 as

$$f_{\text{Castro}}(R_q) = \frac{s(A_q \Rightarrow \text{Class } C_q)}{|\text{Class } C_q| \times |\text{Class } \overline{C_q}|} \times \left(\frac{|\text{Class } \overline{C_q}|}{m} - s(A_q \Rightarrow \text{Class } \overline{C_q}) \right). \quad (19)$$

4. Computer Simulations

In our computer simulations, we used four data sets in Table 1 available from the UCI ML repository. For comparison, some reported results in the literature are also included in Table 1 where the average error rates on test data and the average number of fuzzy rules are cited from [5], [19]–[21]. In the Wisconsin breast cancer data, 16 samples with missing values (among 699 samples in total) were not used in our computer simulations of this

paper. All attribute values of the four data sets were normalized into real numbers in the unit interval $[0, 1]$ before extracting fuzzy rules.

Table 1. Data sets used in this paper and some reported results on those data sets in the literature.

Data set	Number of Attributes	Number of Samples	Number of Classes	Error Rate by C4.5		Fuzzy Rule Base	
				Rel 7	Rel 8	Error Rate	# of Rules
Glass	9	214	6	32.1 [19]	32.5 [19]	42.1 [20]	8.5 [20]
Wisconsin	9	683	2	5.29 [19]	5.26 [19]	4.65 [20]	5.1 [20]
Wine	13	178	3	-	-	3.24 [5]	5.2 [5]
Sonar	60	208	2	28.4 [19]	25.6 [19]	5.77 [21]	125 [21]

Since we did not know an appropriate fuzzy partition for each attribute of each test problem, we simultaneously used four different fuzzy partitions in Figure 3. One of the 14 triangular fuzzy sets was used as an antecedent fuzzy set. For generating simple fuzzy rules (i.e., short fuzzy rules with a small number of antecedent conditions), we also used “*don’t care*” as an antecedent fuzzy set. The membership function of “*don’t care*” is defined as $\mu_{\text{don't care}}(x) = 1$ for $\forall x$ because “*don’t care*” is compatible with any input values. Since each antecedent fuzzy set may assume one of the 14 triangular fuzzy sets in Figure 3 or “*don’t care*”, the total number of combinations of antecedent fuzzy sets is 15^n for an n -dimensional problem. Our task is to find a small number of comprehensible fuzzy rules with high classification ability from 15^n possible rules. In our computer simulations on the sonar data, we only examined fuzzy rules with two or less antecedent conditions (i.e., with $(n - 2)$ or more “*don’t care*” conditions). For the other data sets, we examined fuzzy rules with three or less antecedent conditions. The restriction on the number of antecedent conditions is for finding short (i.e., comprehensible) fuzzy rules as well as for decreasing CPU time.

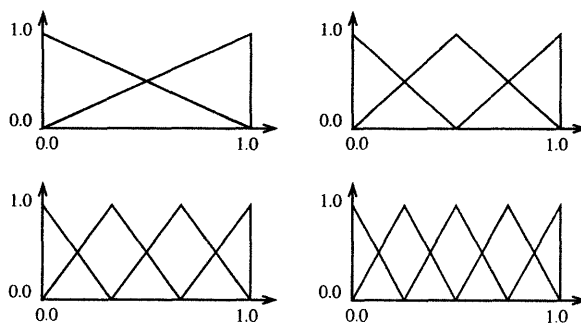


Figure 3. Four fuzzy partitions used in computer simulations.

We extracted N fuzzy rules ($N = 1, 2, \dots$) for each class in each data set using one of the seven rule selection criteria described in the previous section: the confidence, the support, their product, the confidence with the minimum support level, the support with the minimum confidence level, the SLAVE criterion, and the Castro criterion. Several values of

the minimum support and confidence levels were examined for each data set. The rule extraction was performed for each class in a simple greedy manner. First, the best fuzzy rule with respect to a rule selection criterion was found for each class. Next, the second best fuzzy rule was found. In this manner, the best N fuzzy rules were found for each class. There were many cases where multiple fuzzy rules had the best value of a rule selection criterion. In those cases, the tiebreak was performed by applying the following two-step procedure to the multiple fuzzy rules with the best value of the primary criterion (i.e., one of the seven rule selection criteria). The first tiebreak criterion was the number of antecedent conditions. The fuzzy rule with the least antecedent conditions was chosen. This tiebreak criterion is to favor simpler fuzzy rules. When a single fuzzy rule could not be chosen by the first tiebreak criterion, we used the total area of the antecedent fuzzy sets of each fuzzy rule as the second tiebreak criterion. The total area was calculated by simply summing up the area of the triangular membership function of each antecedent fuzzy set in each fuzzy rule. The fuzzy rule with the largest total area was chosen from the competitive fuzzy rules with the same value of the primary criterion and the same number of antecedent conditions. The second tiebreak criterion is to favor more general fuzzy rules that cover larger subspaces of the pattern space. When multiple fuzzy rules still had the same best evaluation with respect to all the three criteria (i.e., the primary criterion and the two tiebreak criteria), a single rule was randomly selected from those best rules. This two-step tiebreak process was used together with each of the seven rule selection criteria.

In order to evaluate the classification performance on test data (i.e., generalization ability), we used the ten-fold cross-validation (10-CV) technique [22] for all the four data sets. Since the classification rate estimated by the 10-CV technique usually depends on the data partition into ten subsets, we executed the whole 10-CV procedure five times using different data partitions. Average error rates on test data estimated by the 10-CV technique are summarized in Table 2 ~ Table 5. Note that the error rate in those tables includes the rejection rate (i.e., the error rate was calculated as $(1 - r_c) \times 100\%$ where r_c is the correct classification rate). When we used the minimum support or confidence level in the rule extraction, there were several cases where a pre-specified number of fuzzy rules could not be generated for some minority classes (because many fuzzy rules did not satisfy the given minimum level). In those cases, the number of actually extracted fuzzy rules was not the same as the number of fuzzy rules specified in the first column of each table.

From these tables, we can see that better results were obtained in many cases from the three criteria based on both the confidence and the support than their individual use (i.e., the second and third columns). In general, the confidence criterion tends to choose fuzzy rules that cover only a small number of patterns from the same class. Thus the classification of many patterns is likely to be rejected. On the other hand, the support criterion tends to choose fuzzy rules that cover many patterns from multiple classes. Thus some patterns are likely to be misclassified. The point is to find a good balance between these two tendencies by combining the confidence and the support into a single rule selection criterion. The SLAVE and Castro criteria can be viewed as attempts for finding such a good balance.

From the comparison of Table 2 ~ Table 5 with Table 1, we can see that a simple greedy method for rule selection is not comparable to the reported results in the literature except for the case of the Wisconsin breast cancer data. The main drawback of the simple greedy method is that the interaction among selected fuzzy rules is not taken into account. This drawback was partially resolved in the iterative fuzzy GBML algorithms [4]-[6] by removing training patterns that had already been covered by the previously found rules.

Table 2. Average error rates on test data of the glass data set.

Number of rules	c	s	Product	c with minimum s -level			s with minimum c -level			SLAVE	Castro
				$s: 0.1$	$s: 0.2$	$s: 0.3$	$c: 0.6$	$c: 0.7$	$c: 0.8$		
6	94.95	51.40	47.85	66.26	73.55	74.81	49.58	42.24	70.28	46.12	48.88
12	94.11	48.46	46.36	59.44	66.82	68.60	48.83	41.12	68.69	45.56	47.90
18	93.36	48.18	47.34	58.27	65.19	67.24	48.08	41.03	67.57	45.98	47.52
24	92.85	47.76	46.92	54.95	62.20	64.21	47.99	40.89	66.73	46.07	48.41
30	92.15	48.04	46.36	54.58	61.68	63.74	47.57	40.84	65.79	46.54	48.08
60	89.63	45.98	46.07	54.49	60.51	62.06	46.54	41.31	63.64	46.17	47.29

Table 3. Average error rates on test data of the Wisconsin breast cancer data set.

Number of rules	c	s	Product	c with minimum s -level			s with minimum c -level			SLAVE	Castro
				$s: 0.1$	$s: 0.2$	$s: 0.3$	$c: 0.6$	$c: 0.7$	$c: 0.8$		
2	90.76	9.78	6.59	51.23	51.42	33.09	9.84	9.08	7.35	6.28	5.70
4	87.07	10.34	7.01	42.46	42.46	23.35	10.28	8.30	8.13	6.85	5.90
6	84.85	8.95	7.07	37.92	37.73	18.67	9.11	5.62	7.09	6.98	4.71
8	83.37	6.18	5.94	33.03	32.88	14.96	5.99	5.46	6.87	6.21	4.58
10	82.46	5.20	5.42	28.57	28.96	12.53	5.20	5.46	6.62	5.21	4.48
60	76.65	4.25	3.97	15.58	16.34	4.44	4.19	4.47	4.55	4.00	4.01

Table 4. Average error rates on test data of the wine data set.

Number of rules	c	s	Product	c with minimum s -level			s with minimum c -level			SLAVE	Castro
				$s: 0.1$	$s: 0.2$	$s: 0.3$	$c: 0.6$	$c: 0.7$	$c: 0.8$		
3	73.60	30.22	11.46	72.70	74.33	74.21	15.39	12.87	10.96	11.35	16.29
6	66.01	22.92	10.84	66.07	66.57	66.46	13.71	11.40	7.25	7.53	12.53
9	52.75	15.11	7.92	35.79	38.88	39.10	14.78	9.72	6.52	7.13	10.11
12	49.89	15.45	6.52	30.17	33.20	33.71	14.49	8.88	6.91	7.25	8.09
15	47.08	15.79	6.57	27.81	25.39	26.18	13.54	7.47	6.69	7.47	7.53
60	44.61	11.18	6.91	5.56	5.84	5.11	6.52	6.69	6.80	6.18	7.13

Table 5. Average error rates on test data of the sonar data set.

Number of rules	c	s	Product	c with minimum s -level			s with minimum c -level			SLAVE	Castro
				$s: 0.1$	$s: 0.2$	$s: 0.3$	$c: 0.6$	$c: 0.7$	$c: 0.8$		
2	98.37	46.63	46.83	97.84	98.03	98.41	26.68	27.55	43.89	27.55	27.60
4	96.49	46.63	47.69	94.38	96.59	96.54	26.78	27.84	38.99	27.40	27.50
6	95.10	46.78	47.45	78.89	93.46	92.50	27.50	27.55	36.59	26.78	27.50
8	94.52	46.78	47.12	68.94	88.89	88.56	27.98	26.54	34.76	26.39	27.79
10	94.09	47.07	45.48	64.47	85.05	83.99	27.98	26.35	33.85	26.35	27.40
60	87.55	46.35	43.03	51.68	47.84	48.17	27.16	25.19	26.35	23.75	25.10

5. Genetic Algorithm-Based Rule Selection

In the above computer simulations, we did not take into account the combinatorial effect of generated fuzzy rules. Thus we could not always find good rule sets. We will be able to find better rule sets with higher classification performance by directly searching for rule sets (i.e., combinations of fuzzy rules) as in many fuzzy GBML algorithms (e.g., [3], [23], [24]). Such a fuzzy GBML algorithm usually requires long CPU time for finding good rule sets for high-dimensional problems. A promising idea for efficiently finding good rule

sets is to search for good subsets of fuzzy rules generated by a heuristic rule selection criterion [7], [8]. In this section, we demonstrate that good rule sets can be obtained as subsets of fuzzy rules generated in the previous section.

In our computer simulations, we used 60 fuzzy rules generated by the product criterion as candidate rules in rule selection. A subset S of those 60 fuzzy rules was handled as an individual and represented by a binary string of the length 60. A genetic algorithm was used for finding the best subset with respect to the following fitness function:

$$\text{fitness}(S) = w_1 \cdot f_1(S) - w_2 \cdot f_2(S) - w_3 \cdot f_3(S), \quad (20)$$

where $f_1(S)$ is the number of correctly classified training patterns by S , $f_2(S)$ is the number of fuzzy rules in S , $f_3(S)$ is the total number of antecedent conditions in S , and w_i is a positive weight for the i -th objective $f_i(S)$. In our computer simulations, the weight values were specified as $w_1 = 100$, $w_2 = 10$ and $w_3 = 1$.

As in the previous section, we executed the whole 10-CV procedure five times for calculating average error rates on test data. Simulation results were summarized in Table 6. From this table, we can see that the classification ability of the extracted fuzzy rules was improved by searching for good combinations of fuzzy rules by genetic algorithms. At the same time, the number of fuzzy rules was significantly decreased in Table 6. This means that the interpretability of fuzzy rule-based systems was improved.

Table 6. Effect of genetic algorithm-based rule selection.

Data set	Before rule selection		After rule selection	
	Error rate	# of rules	Error rate	# of rules
Glass	46.07	60	40.70	7.47
Wisconsin	3.97	60	3.91	4.57
Wine	6.91	60	6.52	5.11
Sonar	43.03	60	29.52	3.68

6. Conclusion

In this paper, we examined the performance of fuzzy rules extracted from numerical data using heuristic rule selection criteria through computer simulations on four data sets with many continuous attributes. Simulation results showed that better results were obtained from composite criteria of the confidence and the support than their individual use. It was also shown that the SLAVE and Castro criteria also worked well. Among the four data sets, we obtained very good results only for the Wisconsin breast cancer data set by a simple greedy method based on heuristic rule selection criteria. Finally we showed that the classification performance of extracted fuzzy rules was improved by searching for their good subsets by genetic algorithms. This suggests that the combinatorial effect of fuzzy rules (i.e., interaction among fuzzy rules) should be taken into account when we design fuzzy rule-based systems.

References

- [1] J. Casillas, O. Cordon, F. Herrera, and L. Magdalena (Eds.), *Trade-off between Accuracy and*

Interpretability in Fuzzy Rule-Based Modelling, Physica-Verlag, 2002.

- [2] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (Eds.), *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, Menlo Park, 1996.
- [3] H. Ishibuchi, T. Nakashima, and T. Murata, "Three-objective genetics-based machine learning for linguistic rule extraction," *Information Sciences*, vol. 136, no. 1-4, pp. 109-133, August 2001.
- [4] A. Gonzalez and R. Perez, "SLAVE: A genetic learning system based on an iterative approach," *IEEE Trans. on Fuzzy Systems*, vol. 7, no. 2, pp. 176-191, April 1999.
- [5] L. Castillo, A. Gonzalez, and R. Perez, "Including a simplicity criterion in the selection of the best rule in a genetic fuzzy learning algorithm," *Fuzzy Sets and Systems*, vol. 120, no. 2, pp. 309-321, June 2001.
- [6] L. Castro, J. J. Castro-Schez, and J. M. Zurita, "Use of a fuzzy machine learning technique in the knowledge acquisition process," *Fuzzy Sets and Systems*, vol. 123, no. 3, November 2001.
- [7] H. Ishibuchi and T. Yamamoto, "Fuzzy rule selection by data mining criteria and genetic algorithms," *Proc. of 2002 Genetic and Evolutionary Computation Conference*, pp. 399-406, July 2002.
- [8] H. Ishibuchi and T. Yamamoto, "Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining," *Fuzzy Sets and Systems* (to appear).
- [9] H. Ishibuchi, K. Nozaki, and H. Tanaka, "Distributed representation of fuzzy rules and its application to pattern classification," *Fuzzy Sets and Systems*, vol. 52, no. 1, pp. 21-32, November 1992.
- [10] J. van den Berg, U. Kaymak, and W. -M. van den Bergh, "Fuzzy classification using probability based rule weighting," *Proc. of 11th IEEE International Conference on Fuzzy Systems*, pp. 991-996, May 2002.
- [11] T. -P. Hong, C. -S. Kuo, and S. -C. Chi, "Trade-off between computation time and number of rules for fuzzy mining from quantitative data," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 5, 587-604, October 2001.
- [12] H. Ishibuchi, T. Yamamoto, and T. Nakashima, "Fuzzy data mining: Effect of fuzzy discretization," *Proc. of 1st IEEE International Conference on Data Mining*, pp.241-248, November 2001.
- [13] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," *Proc. of 20th International Conference on Very Large Data Bases*, pp. 487-499, 1994. Expanded version is available as IBM Research Report RJ9839, 1994.
- [14] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo, "Fast discovery of association rules," in U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (eds.) *Advances in Knowledge Discovery and Data Mining*, pp. 307-328, AAAI Press, Metro Park, 1996.
- [15] H. Ishibuchi and T. Yamamoto, "Comparison of heuristic rule weight specification methods," *Proc. of 11th IEEE International Conference on Fuzzy Systems*, pp. 908-913, May 2002.
- [16] H. Ishibuchi and T. Nakashima, "Effect of rule weights in fuzzy rule-based classification systems," *IEEE Trans. on Fuzzy Systems*, vol. 9, no. 4, pp. 506-515, August 2001.
- [17] H. Ishibuchi, T. Nakashima, and T. Morisawa, "Voting in fuzzy rule-based systems for pattern classification problems," *Fuzzy Sets and Systems*, vol. 103, no. 2, pp. 223-238, April 1999.
- [18] O. Cordon, M. J. del Jesus, and F. Herrera, "A proposal on reasoning methods in fuzzy rule-based classification systems," *International Journal of Approximate Reasoning*, vol. 20, no. 1, pp. 21-45, January 1999.
- [19] J. R. Quinlan, "Improved use of continuous attributes in C4.5," *Journal of Artificial Intelligence Research*, vol. 4, pp. 77-90, March 1996.
- [20] L. Sanchez, I. Couso, and J. A. Corrales, "Combining GA operators with SA search to evolve fuzzy rule base classifiers," *Information Sciences*, vol. 136, no. 1-4, pp. 175-191, August 2001.
- [21] J. Casillas, O. Cordon, M. J. del Jesus, F. Herrera, "Genetic feature selection in a fuzzy rule-based classification system learning process for high-dimensional problems," *Information Sciences*, vol. 136, no. 1-4, pp. 135-157, August 2001.
- [22] S. M. Weiss and C. A. Kulikowski, *Computer Systems That Learn*, Morgan Kaufmann Publishers, San Mateo, 1991.
- [23] H. Ishibuchi, T. Nakashima, and T. Murata, "Performance evaluation of fuzzy classifier systems for multi-dimensional pattern classification problems," *IEEE Trans. on Systems, Man, and Cybernetics- Part B: Cybernetics*, vol. 29, no. 5, pp. 601-618, October 1999.
- [24] H. Ishibuchi and T. Nakashima, "Improving the performance of fuzzy classifier systems for pattern classification problems with continuous attributes," *IEEE Trans. on Industrial Electronics*, vol. 46, no. 6, pp. 157-168, December 1999.

Linguistic Hedges: a Quantifier Based Approach

Martine DE COCK

*Dept. of Applied Mathematics and Computer Science
 Ghent University, Krijgslaan 281 (S9), B-9000 Gent, Belgium
 Martine.DeCock@rug.ac.be, <http://fuzzy.rug.ac.be>*

Abstract. We present an entirely new approach for the representation of intensifying and weakening linguistic hedges in fuzzy set theory, which is primarily based on a crisp ordering relation associated with the term that is modified, as well as on a fuzzy quantifier. With this technique we can generate membership functions for both atomic and modified linguistic terms. We prove that our model respects semantic entailment and we show that it surpasses traditional approaches, such as powering and shifting modifiers, on the intuitive level and on the level of applicability.

1 Introduction

The success of fuzzy expert systems is greatly due to their ability to represent and handle vague information expressed by means of linguistic terms in facts and (IF-THEN) rules. Indeed, this tends to make such systems very compact and tolerant for the imprecision and incompleteness that often afflicts our knowledge of the real world. Furthermore, their input, output and inference mechanism can be more easily understood by humans, who's most popular daily means of communication and reasoning is, after all, natural language. In fuzzy systems, each linguistic term is represented by a fuzzy set on a suitable universe X [14]. A fuzzy set A on X is a $X \rightarrow [0, 1]$ mapping, also called the membership function of A , such that for all x in X , $A(x)$ is the membership degree of x in A . In this paper we will use the same notation to denote a fuzzy set A and the term A represented by it.

One of the most crucial and often difficult tasks in developing a fuzzy expert system is the construction of the membership functions for the linguistic terms involved. Fortunately these terms have a specific structure [15] which allows to partially automate this task by computing the membership functions of composed linguistic terms from those of atomic ones. In this context, an atomic linguistic term is an adjective (e.g. expensive). Composed terms are generated by applying either a linguistic modifier to a term (e.g. very expensive), negating a term (e.g. not very expensive) or by combining terms by means of a connective (e.g. not very expensive or sophisticated, nice and easy). If A and B are fuzzy sets on X , the terms A and B , A or B and not A can respectively be modelled by the Zadeh-intersection, the Zadeh-union and the complement, defined as follows

$$\begin{aligned} A \cap B(x) &= \min(A(x), B(x)) \\ A \cup B(x) &= \max(A(x), B(x)) \\ co A(x) &= 1 - A(x) \end{aligned}$$

for all x in X . It is even more common to model them by \mathcal{T} -intersection, \mathcal{S} -union and \mathcal{N} -complement based on a triangular norm \mathcal{T} , a triangular conorm \mathcal{S} and a negator \mathcal{N} ; the above mentioned operations are just special cases of this more general approach. In this paper however we will focus on linguistic modifiers (also called linguistic hedges). Since the origin of fuzzy set theory, many researchers have paid attention to the representation of these adverbs, probably because they allow for the generation of many modified terms from existing ones. The first serious attempt was sketched by Zadeh, as early as 1972 [15]. For the representation of very A and more or less A , he defined the concentration and dilation operator (also called powering modifiers) based on a simple involution, i.e. very $A(x) = A(x)^2$ and more or less $A(x) = A(x)^{0.5}$, for all x in X . One can easily verify that in this representation

$$\text{very } A(x) \leq A(x) \leq \text{more or less } A(x)$$

for all x in X . The satisfaction of this so-called semantical entailment [11] (very A is a subset of A , which in turn is a subset of more or less A) can be considered as a prerequisite for any representation, since very is an intensifying modifier, while more or less has a weakening effect. The best known shortcomings of Zadeh's approach, pointed out in e.g. [8, 9, 11], are that they keep the kernel and the support, which are defined as

$$\begin{aligned} \ker A &= \{x|x \in X \wedge A(x) = 1\} \\ \text{supp } A &= \{x|x \in X \wedge A(x) > 0\} \end{aligned}$$

As a consequence, this representation cannot distinguish between being A to degree 1 and being very A to degree 1. One might feel however that a person of 80 years is old to degree 1 but very old only to a lower degree (e.g. 0.7), but this cannot be modelled by means of powering modifiers. The shifting modifiers, informally suggested by Lakoff [11] and more formally developed later on [3, 8, 9], do not have this shortcoming, but it is a serious drawback that they cannot be applied straightforwardly to all kinds of membership functions in the same way, and hence sometimes require artificial tricks. Many representations developed in the same period are afflicted with very similar disadvantages as the powering and shifting modifiers (we refer to [10] for an overview). We believe these kinds of shortcomings on the level of intuition and the level of applicability are due to the fact that these modifiers are only technical tools, lacking inherent meaning.

In fact it was not until the second half of the 1990's that new models with a clear semantics started to surface. In the horizon approach [12] the semantics is derived from the concepts of horizon and visibility [5]. It is one of the few techniques in which the representation for both atomic and modified terms is generated from within the model (which can be a constraint if one prefers to modify membership functions for atomic terms obtained elsewhere). In the fuzzy relational based model [6], the semantics is retrieved from the context. A characteristic of the traditional approaches is that they do not really look at the context: to determine the degree to which x is very A , the concentration operator for instance only looks at the degree to which x is A and ignores the objects in the context of x . By bringing a fuzzy relation into action however, the membership degrees of the elements related to x can also be taken into account to some extent. Depending on the nature of the fuzzy relation, different kind of linguistic modifiers can be modelled within the same framework: an ordering relation gives rise to ordering-based modifiers such as at most and at least [2], while a resemblance relation can be used to represent both weakening (such as roughly and more or less) and intensifying hedges (such as very and extremely) [4]. Informal starting points for the latter two are observations such as "a person is more or less old if he resembles an old person" and "a person is

very old if everybody he resembles is old". Translating all components of these observations into their fuzzy set theoretical counterparts, results on the formal level in the use of the direct image (related to the compositional rule of inference) and superdirect image of the fuzzy sets being modified under the fuzzy relations that describe the context.

In this paper we will once again attempt the technique of transforming natural language statements, describing the meaning of (modified) terms, into their mathematical counterparts. This time however the starting point will not be our intuition, but an analysis made by Wheeler in 1972 [13]. Putting all pieces of the puzzle together in the right way gives rise to a stunningly elegant, computationally efficient and semantically very comprehensible representation of both (!) atomic terms and modified terms. The key notion is that of a quantifier, which is not surprising since Wheeler's goal was to reveal that English is a first-order language at some level of analysis. Besides this, relations are once again brought into action to model the context. Most remarkable (compared to the fuzzy relational based approach) is however that crisp *ordering* relations prove to be very useful to model intensifying and weakening hedges. Since Wheeler describes the meaning of *rather* and *very*, we will study the representation of precisely these two hedges. However before presenting the model for modified terms (Section 4), we go into the representation of atomic terms (Section 3) after the necessary preliminaries (Section 2).

2 Preliminaries

Throughout this paper, let X denote a finite universe of discourse (i.e. a non-empty set containing a finite number of objects we want to say something about). In the following, the class of all fuzzy sets on X will be denoted by $\mathcal{F}(X)$. A fuzzy set A takes membership values in the real unit interval $[0, 1]$. If all membership values of A belong to $\{0, 1\}$, A is called a crisp set. In this case, the notation $A(x) = 1$ corresponds to $x \in A$, while $A(x) = 0$ is the same as $x \notin A$. A fuzzy relation R on X is a fuzzy set on $X \times X$. If the relation R is crisp, we denote $(x, y) \in R$ also by xRy . A useful concept concerning fuzzy relations is that of *foreset*.

Definition 1 (Foreset). [1] Let R be a fuzzy relation on X and y in X . The R -foreset of y is the fuzzy set Ry defined by $Ry(x) = R(x, y)$, for all x in X .

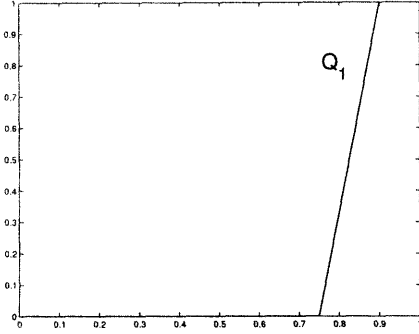
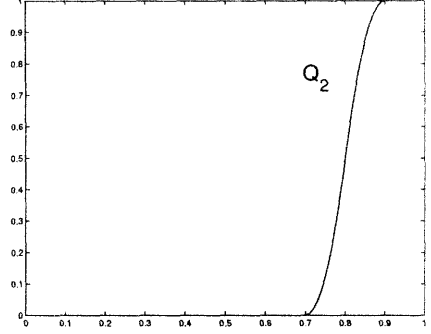
In other words the R -foreset of y is the fuzzy set of objects related to y . Furthermore we need the concept of inclusion of fuzzy set, as a means to express the mathematical counterpart of semantic entailment.

Definition 2 (Inclusion). For A and B in $\mathcal{F}(X)$ one says that A is included in B , denoted $A \subseteq B$, if and only if $A(x) \leq B(x)$, for all x in X .

As already mentioned in the introduction a quantifier will play a crucial role in our model. Intuitively, quantifiers relate to the concept of cardinality of sets. One can easily verify that the power of a fuzzy set is a generalization of the classical concept of cardinality of a crisp set.

Definition 3 (Power). [7] The power of a fuzzy set A on X is defined as

$$|A| = \sum_{x \in X} A(x)$$

Figure 1: Fuzzy quantifier Q_1 Figure 2: Fuzzy quantifier Q_2

More specifically we need a kind of relative cardinality, called proportion.

Definition 4 (Proportion). [17] For A and B fuzzy sets on X , the (size of the) relative proportion of A in B is given by

$$\frac{|A \cap B|}{|A|}$$

Definition 5 (Fuzzy quantifier). [17] A fuzzy quantifier Q is a $[0, 1] - [0, 1]$ mapping. Q is called regular increasing if it is increasing (i.e. $x \leq y \Rightarrow Q(x) \leq Q(y)$ for all x and y in $[0, 1]$) and if it satisfies the boundary conditions $Q(0) = 0$ and $Q(1) = 1$.

Figures 1 and 2 depict two regular increasing fuzzy quantifiers. Finally we recall that Zadeh [18] suggested to calculate the truth value of

$$Q \text{ } A\text{'s are } B\text{'s} \quad (1)$$

as

$$Q \left(\frac{|A \cap B|}{|A|} \right)$$

3 Representation of atomic terms

Wheeler [13] suggests to view adjectives as relations between an individual and a class of individuals. A possibility is to analyze “John is a tall man”, which means “John is tall for a man”, as “John is taller than most men”. In this way the meaning of the *vague* linguistic term tall is distributed over a *crisp* ordering relation (is taller than) and a quantifier (most) that helps to preserve the vagueness of the original expression. To obtain something of the form (1) we suggest to further analyze this as “most men are shorter than John”.

More formally, to compute the membership degrees for a term A , we suggest to associate with it a crisp ordering relation R (i.e. a reflexive, anti-symmetrical and transitive relation) on the universe X which ranks all elements of X in descending order of satisfying A . From now on we will refer to this relation as “the relation associated with the term”. E.g. when representing the term young we will use the relation “is older than or equal to”, for expensive we can use “is cheaper than or equal to” etc. For y in X , the foreset Ry then denotes the set of objects related to y , e.g. the set of ages older than y , the set of prices cheaper than y etc. The

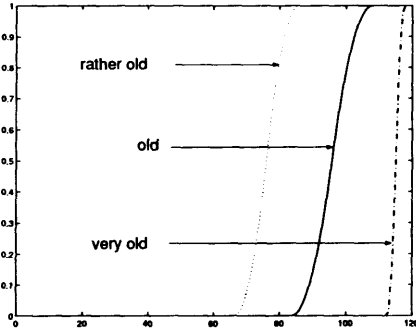


Figure 3: Old and modified terms

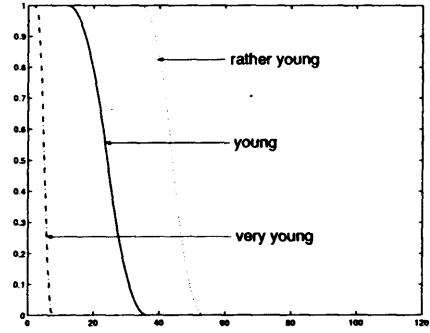


Figure 4: Young and modified terms

membership degree of y in A can now be computed as the degree to which most elements of the universe are related to y , i.e. the truth degree of “most X ’s are Ry ”, which leads us to the following representational scheme:

Scheme 1. Let Q be a regular increasing quantifier representing most. If R is the ordering relation associated with a term, this term can be represented by the fuzzy set A defined as

$$A(y) = Q\left(\frac{|Ry|}{|X|}\right)$$

for all y in X .

Proposition 1. If A is constructed as described in Scheme 1, then A is increasing w.r.t. the ordering R , i.e.

$$xRy \Rightarrow A(x) \leq A(y)$$

for all x and y in X .

Proof. For all u in X it holds: if $u \in Rx$ then uRx . Because of the transitivity of R and the assumption xRy also uRy or $u \in Ry$. Hence $Rx \subseteq Ry$ and $|Rx| \leq |Ry|$. The proposition now follows from Scheme 1 and the increasing nature of Q . \square

Example 1. The membership functions for the atomic terms old, young and middle – aged in the universe of ages $X = \{0, 1, \dots, 120\}$ as depicted in Figures 3, 4 and 5 were generated using the fuzzy quantifier Q_2 of Figure 2. Furthermore we respectively used the ordering relations R_1 , R_2 and R_3 defined by

$$\begin{aligned} (x, y) \in R_1 & \text{ if and only if } x \leq y \\ (x, y) \in R_2 & \text{ if and only if } y \leq x \\ (x, y) \in R_3 & \text{ if and only if } |y - 60| \leq |x - 60| \end{aligned}$$

for all x and y in X , with \leq being the usual ordering on natural numbers.

4 Representation of modified terms

Wheeler[13] analyzes “John is a very tall man” as “John is tall for a tall man”. Furthermore he suggests that “a rather tall man is a man who bears the “tall” relation to men who are not tall but who may or may not be one of the men who are not tall”. In other words we could informally say that “John is very tall” if “most tall men are shorter than John” and that “John is rather tall” if “most not tall men are shorter than John”, leading to the following representational scheme:

Scheme 2. Let Q be a regular increasing quantifier representing *most*. If A is the representation of a term and R is the ordering relation associated with this term, the modified terms can be represented by

$$\begin{aligned}\text{very } A(y) &= Q \left(\frac{|A \cap Ry|}{|A|} \right) \\ \text{rather } A(y) &= Q \left(\frac{|co A \cap Ry|}{|co A|} \right)\end{aligned}$$

for all y in X .

The following proposition reveals that these representations are surprisingly efficient from the computational point of view .

Proposition 2. If the representations for *very* A and *very* B are constructed according to Scheme 2 then

$$\begin{aligned}\text{very } A(y) &= Q \left(\frac{\sum_{x \in Ry} A(x)}{\sum_{x \in X} A(x)} \right) \\ \text{rather } A(y) &= Q \left(\frac{\sum_{x \in Ry} (1 - A(x))}{\sum_{x \in X} (1 - A(x))} \right)\end{aligned}$$

Proof. The first equality is an immediate result of

$$\frac{|A \cap Ry|}{|A|} = \frac{\sum_{x \in X} \min(A(x), Ry(x))}{\sum_{x \in X} A(x)} = \frac{\sum_{x \in Ry} A(x)}{\sum_{x \in X} A(x)} \quad (2)$$

The second equality can be proven analogously. □

Note that the denominators in both expressions do not depend on y and therefore have to be computed only once. Furthermore if the membership degrees are computed in the ordering implied by R , the numerator in the expression for an element y can be obtained simply by adding one term to the numerator of the expression for the preceding element. This new term is based on the membership degree of y in A (in the case of *very* it is precisely $A(y)$, in the case of *rather* it is $1 - A(y)$), which illustrates very nicely how the context is expanded gradually with every new element taken into account.

In the next theorem we show that the representations of Scheme 1 and 2 satisfy semantic entailment. For better understanding of the proof, note that for all strictly positive real numbers a, b, c and d such that $a \leq d$ and $b \leq c$, one can easily verify that

$$\frac{a+b}{a+c} \leq \frac{d+b}{d+c}$$

I.e. if the denominator of a fraction is dominating, replacement of identical terms in numerator and denominator by larger ones, gives rise to an enlargement of the fraction as a whole.

Theorem 1 (Inclusiveness). If R is a total ordering then

$$\text{very } A \subseteq A \subseteq \text{rather } A$$

Proof. Replacing $A(x)$ by its definition in (2) we obtain

$$\frac{|A \cap Ry|}{|A|} = \frac{\sum_{x \in Ry} Q(\frac{|Rx|}{|X|})}{\sum_{x \in X} Q(\frac{|Rx|}{|X|})} = \frac{\sum_{x \in Ry} Q(\frac{|Rx|}{|X|})}{\sum_{x \in Ry} Q(\frac{|Rx|}{|X|}) + \sum_{x \notin Ry} Q(\frac{|Rx|}{|X|})}$$

If $x \in Ry$, the transitivity of R implies $Rx \subseteq Ry$. (Indeed, for every $u \in Rx$ it holds that $R(u, x) = 1$. Furthermore because of the assumption that $x \in Ry$ also $R(x, y) = 1$ holds and hence by the transitivity of R we obtain $R(u, y) = 1$, in other words $u \in Ry$.) Therefore, if $x \in Ry$, $|Rx| \leq |Ry|$ holds. Since Q is increasing we obtain

$$\frac{|A \cap Ry|}{|A|} \leq \frac{|Ry| \cdot Q(\frac{|Ry|}{|X|})}{|Ry| \cdot Q(\frac{|Ry|}{|X|}) + \sum_{x \notin Ry} Q(\frac{|Rx|}{|X|})}$$

If $x \notin Ry$ by a similar reasoning as above (assuming that R is a total ordering) we obtain $|Ry| \leq |Rx|$ and hence

$$\frac{|A \cap Ry|}{|A|} \leq \frac{|Ry| \cdot Q(\frac{|Ry|}{|X|})}{|Ry| \cdot Q(\frac{|Ry|}{|X|}) + (|X| - |Ry|)Q(\frac{|Ry|}{|X|})} = \frac{|Ry|}{|X|}$$

The first inclusion now follows immediately from Scheme 1 and 2 and the fact that Q is increasing. The second inclusion can be proven analogously. \square

Example 2. Figures 3, 4 and 5 show membership functions for modified terms generated using the same fuzzy quantifier and the same ordering relations as in Example 1. In Figure 6 the same exercise was done using fuzzy quantifier Q_1 of Figure 1. It is clear from all these figures that the proposed representational schemes respect semantic entailment as shown in Theorem 1, and that the kernel and the support are changed in the modification process. Furthermore the figures illustrate that the same technique can be straightforwardly applied for increasing, decreasing and partially increasing and decreasing membership functions.

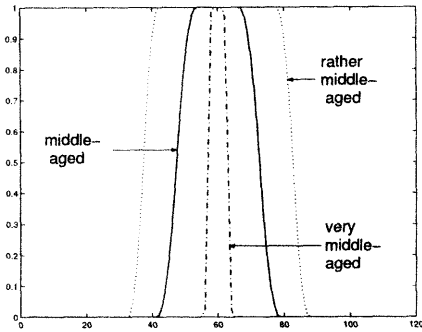


Figure 5: Middle-aged and modified terms

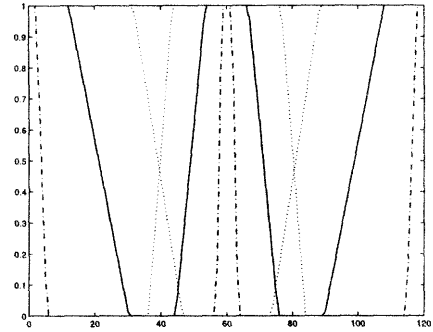


Figure 6: Membership functions in the universe of ages

5 Conclusion

We have presented a new approach for the fuzzy set theoretical representation of both atomic terms and terms modified by the intensifying hedge *very* and the weakening hedge *rather*. Our model is based on a crisp ordering relation that is closely related to the meaning of the term being represented, and that helps to take the context into account. The vagueness of the term is preserved by the use of a fuzzy quantifier. The model turns out to be efficient from the computational point of view, and it respects semantic entailment ($\text{very } A \subseteq A \subseteq \text{rather } A$). Finally it clearly surpasses traditional approaches, such as powering and shifting modifiers, on the intuitive level and the level of applicability, which is probably due to its clear inherent semantics.

Acknowledgement

The author would like to thank the Fund for Scientific Research - Flanders (FWO) for funding the research reported on in this paper.

References

- [1] W. Bandler and L. J. Kohout, Fuzzy Relational Products as a Tool for Analysis and Synthesis of the Behaviour of Complex Natural and Artificial Systems, in: S. K. Wang and P. P. Chang (Ed.), *Fuzzy Sets: Theory and Application to Policy Analysis and Information Systems*, Plenum Press, New York and London (1980) 341–367.
- [2] U. Bodenhofer, A Similarity-Based Generalization of Fuzzy Orderings, *Schriftenreihe der Johannes-Kepler-Universität Linz*, C 26, Universitätsverlag Rudolf Trauner (1999).
- [3] B. Bouchon-Meunier, *La Logique Floue*, series: Que sais-je?, Vol. 2702, Paris (1993).
- [4] M. De Cock and E. E. Kerre, A New Class of Fuzzy Modifiers, in: *Proceedings ISMVL 2000 (30th IEEE International Symposium on Multiple-Valued Logic, IEEE Computer Society, (2000) 121–126*.
- [5] M. De Cock, Fuzzy Hedges: a Next Generation, in: (Kristina Striegnitz, ed.) *Proceedings of the ESS-LLI2001 Student Session*. (2001) 59–70.
- [6] M. De Cock, E. E. Kerre, Fuzzy Modifiers Based on Fuzzy Relations, to appear in *Information Sciences* (2002).

- [7] A. De Luca, S. Termini, A Definition of Non-probabilistic entropy in the setting of fuzzy sets, *Inform. Control* **20** (1972) 301–312.
- [8] H. Hellendoorn, Reasoning with fuzzy logic, Ph. D. thesis, T.U. Delft (1990).
- [9] E. E. Kerre, Introduction to the Basic Principles of Fuzzy Set Theory and Some of its Applications, Communication and Cognition, Gent (1993).
- [10] E. E. Kerre, M. De Cock, Linguistic Modifiers: an overview, in: G. Chen, M. Ying and K.-Y. Cai (Ed.), *Fuzzy Logic and Soft Computing*. Kluwer Academic Publishers (1999) 69–85.
- [11] G. Lakoff, Hedges : a Study in Meaning Criteria and the Logic of Fuzzy Concepts, *Journal of Philosophical Logic* **2**(1973) 458–508.
- [12] V. Novák, A Horizon Shifting Model of Linguistic Hedges for Approximate Reasoning, in: *Proc. I of the Fifth IEEE International Conference on Fuzzy Systems*, New Orleans (1996) 423–427.
- [13] S. C. Wheeler III, Attributives and their Modifiers, *Noûs* **VI**, **4** (1972) 310–334.
- [14] L. A. Zadeh, Fuzzy Sets, *Information and Control* **8** (1965) 338–353.
- [15] L. A. Zadeh, A Fuzzy-Set-Theoretic Interpretation of Linguistic Hedges, *Journal of Cybernetics*, **2,3** (1972) 4–34.
- [16] L. A. Zadeh, The Concept of a Linguistic Variable and its Application to Approximate Reasoning, part I, II, III, *Information Sciences*, **8** (1975), 199–249, 301–357; **9** (1975) 43–80.
- [17] L. A. Zadeh, A Computational Approach to Fuzzy Quantifiers in Natural Languages, *Comput. Math. Appl.* **9** (1983) 149–184.
- [18] L. A. Zadeh, Test-score Semantics as a Basis for a Computational Approach to the Representation of Meaning, in: (R. R. Yager et. al, Ed.), *Fuzzy Sets and Applications: Selected Papers by L. A. Zadeh*, Wiley, New York (1987) 655–684.

Section 4

Evolutionary Computation

This page intentionally left blank

Analyzing the Founder Effect in Simulated Evolutionary Processes Using Gene Expression Programming

Cândida FERREIRA

*Gepsoft, 37 The Ridings,
Bristol BS13 8NU, UK
candidaf@gepsoft.com*

Abstract. Gene expression programming is a genotype/phenotype system that evolves computer programs encoded in linear chromosomes of fixed length. The interplay between genotype (chromosomes) and phenotype (expression trees) is made possible by the structural and functional organization of the linear chromosomes. This organization allows the unconstrained operation of important genetic operators such as mutation, transposition, and recombination. Although simple, the genotype/phenotype system of gene expression programming can provide some insights into natural evolutionary processes. In this work the question of the initial diversity in evolving populations of computer programs is addressed by analyzing populations undergoing either mutation or recombination. The results presented here show that populations undergoing mutation recover practically undisturbed from evolutionary bottlenecks whereas populations undergoing recombination alone depend considerably on the size of the founder population and are unable to evolve efficiently if subjected to really tight bottlenecks.

Introduction

Everybody agrees that, by and large, evolution relies on genetic variation coupled with some kind of selection and, in fact, all evolutionary algorithms explore these fundamental processes. However, there is no agreement concerning the best way to create genetic variation, with researchers divided between mutation and recombination [1, 4, 6, 7, 8, 9, 12]. This fact *per se* is extremely revealing, suggesting that existing artificial systems are fundamentally different from one another. Particularly interesting is that the evolvability of the system will depend heavily on the kind of genetic operator used to create variation. And the size and kind of initial populations is closely related to this question.

In all evolutionary algorithms, an evolutionary epoch or run starts with an initial population. Initial populations, though, are generated in many different ways, and the performance and the costs (in terms of CPU time) of different algorithms depend greatly on the characteristics of initial populations. The simplest and less time consuming population is the totally random initial population. However, few evolutionary algorithms are able to use this kind of initial population due not only to structural constraints but also to the kind of genetic operators available to create genetic modification. The initial populations of gene expression programming (GEP) are totally random and consist of the linear genomes of the individuals of the population [4].

Gene expression programming is a genotype/phenotype system that evolves computer programs of different sizes and shapes (expression trees) encoded in linear chromosomes of fixed length. The genetic encoding used in GEP allows a totally unconstrained interplay between chromosomes and expression trees. This interplay brought about a tremendous increase in performance allowing, consequently, the undertaking of detailed, much needed analysis of fundamental evolutionary processes.

One such analysis is the importance of the initial diversity in evolution. Ernst Mayr hypothesized that, in nature, small groups of founder individuals can give rise to a new species [10, 11]. This is only possible, however, due to the variety of genetic operators that continually introduce genetic modification in the population. The initial diversity question is extremely important in artificial evolutionary systems where founder events are created each time a run starts. And the efficiency of the system will depend, among other things, on how the system deals with evolutionary bottlenecks. Indeed, the evolutionary strategies followed by different artificial evolutionary systems depend greatly on the nature of their respective initial populations.

In GEP, due to the existence of a truly functional and autonomous genome, the implementation of different genetic operators is extremely simplified and Ferreira [4] introduces seven. Furthermore, due to the high efficiency of the algorithm, the performance and the roles of all these operators can be easily and rigorously analyzed revealing the existence of two fundamental types of evolutionary dynamics: non-homogenizing dynamics found in populations undergoing mutation or other non-conservative operators, and homogenizing dynamics found in populations undergoing recombination alone [5]. Therefore, systems with different evolutionary behaviors can be easily simulated in GEP. In this work, the importance of the initial diversity is analyzed in two different systems. The first evolves under mutation and has a non-homogenizing dynamics characteristic of an efficient adaptation. The second evolves under recombination and has a homogenizing dynamics characteristic of poorly evolving systems.

1. Genetic Algorithms

All genetic algorithms use populations of individuals, select individuals according to fitness, and introduce genetic variation using one or more genetic operators. Structurally, genetic algorithms can be subdivided in three fundamental groups: (1) Genetic algorithms with individuals consisting of linear chromosomes of fixed length devoid of complex expression. In these systems, replicators (chromosomes) survive by virtue of their own properties. The algorithm invented by Holland [8] belongs to this group, and is known as genetic algorithm or GA; (2) Genetic algorithms with individuals consisting of ramified structures of different sizes and shapes and, therefore, capable of assuming a richer number of functionalities. In these systems, replicators (ramified structures) also survive by virtue of their own properties. The algorithm invented by Cramer [2] and later developed by Koza [9] belongs to this group and is known as genetic programming or GP; and (3) Genetic algorithms with individuals encoded as linear chromosomes of fixed length which are afterwards expressed as ramified structures of different sizes and shapes. In these systems, replicators (chromosomes) survive by virtue of causal effects on the phenotype (ramified structures). The algorithm invented by myself [4] belongs to this group and is known as gene expression programming or GEP.

It is worth emphasizing that GEP shares with GP the same kind of ramified structure, meaning that both systems can be used in the same problem domains. However, due to the crossing of the phenotype threshold [3], gene expression programming is bound to be much

more successful, allowing the exploration of new frontiers in evolutionary computation. Below are briefly highlighted some of the differences between GP and GEP.

1.1 Genetic Programming

As simple replicators, the ramified structures of GP are tied up in their own complexity: on the one hand, bigger, more complex structures are more difficult to handle during reproduction and, on the other, the introduction of genetic variation can only be done at the tree level and, therefore, must be done carefully so that valid structures are created. For instance, the tree-specific recombination is practically the only source of genetic variation used in GP for it allows the exchanging of sub-trees and, therefore, always produces valid structures. But the implementation of other operators, like the equivalent of the high-performing natural point mutation, is unproductive as most mutations would have resulted in syntactically incorrect structures.

Obviously, the implementation of other operators such as transposition or inversion raises similar difficulties. In fact, Koza [9] describes two other tree-specific operators, permutation and mutation, but they are seldom used in GP.

1.2 Gene Expression Programming

The phenotype of GEP individuals consists of the same kind of ramified structures used in genetic programming. However, these complex entities are encoded in simpler, linear structures of fixed length – the chromosomes. Thus, there are two main players in GEP: the chromosomes and the ramified structures or expression trees (ETs), the latter being the expression of the genetic information encoded in the former. As in nature, the process of information decoding is called translation. And this translation implies obviously a kind of code and a set of rules. The genetic code is very simple: a one-to-one relationship between the symbols of the chromosome and the functions or terminals they represent. The rules are also very simple: they determine the spatial organization of the functions and terminals in the ETs and the type of interaction between sub-ETs in multigenic systems.

In GEP there are therefore two languages: the language of the genes and the language of ETs. However, thanks to the simple rules that determine the structure of ETs and their interactions, it is possible to infer immediately the phenotype given the sequence of a gene, and *vice versa*. This bilingual and unequivocal system is called *Karva* language. The details of this new language are given in [4].

2. Artificial Evolutionary Systems and the Founder Effect

The question of the initial diversity is pertinent in artificial evolutionary systems for two main reasons. First, the random generation of viable individuals in some complex problems can be a rare event and, in those cases, it would be advantageous if the evolutionary process could get started from one or a few founder individuals; whether this is possible or not, will depend on the modification mechanisms available to the system. And, second, because of this, the kind of mechanism used to create genetic variation becomes of paramount importance. If genetic variation is created by non-homogenizing operators such as point mutation, then populations will be able to adapt and evolve. However, if genetic variation is created by homogenizing

operators (recombination), then evolution is either altogether halted when only one founder individual is available or seriously compromised when the number of founder individuals is excessively small.

The importance of the initial diversity in evolution was stressed by E. Mayr in what he called founder effect speciation [10, 11]. This process may be thought of as the establishment of a new population due to a founder event initiated by genetic drift and followed by natural selection. An extreme case of a founder event is the colonization of a previously uninhabited area by a single pregnant female. In nature, besides recombination, other genetic operators are used to create modification and populations that pass through a bottleneck are capable of adaptation, sometimes even originating new species.

Similarly, in artificial evolutionary systems, the capability of founder populations to evolve depends greatly on the kind of mechanism used to create genetic modification. Indeed, if homogenizing operators are the only source of genetic modification, populations will not be able to evolve efficiently or not at all in the extreme case of only one founder individual.

In this work, different populations of computer programs will be used to analyze the founder effect in evolution. One kind of population uses point mutation as the only source of genetic modification and the other uses only recombination.

3. Setting the System

In order to quantify accurately how different populations respond to the number of actual founders in initial populations, a simple, exactly solved problem must be chosen. This problem must allow the comparison of dissimilarly performing genetic operators, such as the high-performing point mutation and the less powerful recombination. In addition, the populations chosen to make the comparisons must follow different evolutionary dynamics so that the results discussed here could be useful not only theoretically but also for understanding the evolutionary strategies chosen by different artificial evolutionary systems.

3.1 General Settings

To analyze the founder effect on populations undergoing either mutation or recombination, the following test sequence was chosen:

$$a_n = 4n^4 + 3n^3 + 2n^2 + n$$

where n consists of the nonnegative integers. This sequence was chosen for three reasons. First, it can be exactly solved by the algorithm and therefore provide an accurate measure of performance in terms of success rate. Second, it requires relatively small populations and relatively short evolutionary times, making the task feasible. And third, it provides sufficient resolution to allow the comparison of dissimilarly performing operators such as mutation and recombination.

In all the experiments, the first 10 positive integers n and their corresponding term were used as fitness cases; the fitness function was based on the relative error with a selection range of 20% and maximum precision (0% error), giving maximum fitness $f_{\max} = 200$ [4]; the selection was made by roulette-wheel sampling coupled with simple elitism; population sizes P of 50 individuals and evolutionary times $G = 100$ generations were used; the success rate of each experiment was evaluated over 100 independent runs; $F = \{+, -, *, /\}$ and the terminal set T consisted only of the independent variable which was represented by a , giving $T = \{a\}$; and six-genic chromosomes of length 78 (head length $h = 6$) linked by addition were used.

In gene expression programming mutation is by far the single most important genetic operator and populations undergoing mutation display non-homogenizing dynamics [5], i.e., the best fitness is always considerably above average fitness and average fitness displays a pronounced oscillatory pattern. Furthermore, mutation is the only operator capable of reaching the performance peak and, for each experiment, this peak can be found. Figure 1 shows the performance peak for populations evolving using the parameters given above. In this case, maximum performance is reached around a mutation rate $p_m = 0.05$. Therefore, this value will be used to study the importance of the initial diversity in non-homogenizing populations.

As for recombination, this operator is the less powerful of all GEP operators and populations undergoing recombination alone display homogenizing dynamics [5], i.e., with time, the best fitness becomes equal to average fitness and populations lose all genetic diversity. Furthermore, it has been shown that the three kinds of GEP recombination (two-point, one-point and gene recombination) perform better at maximum rates of 1.0, being two-point recombination the most powerful of the three recombinational operators and gene recombination the less powerful [5]. However, for the particular settings used in this analysis, when used separately, the three kinds of recombination perform so poorly that the three recombinational operators were combined together so that the performance of the algorithm increased a little (Table 1, column 5). As shown in the fifth column of Table 1, in this experiment, the recombination rates of the three recombinational operators are identical and equal to 0.8. Note that the success rate increases slightly comparatively to the individual performances obtained for the recombination operators working separately. So, the dependence of success rate on the number of actual founders in populations undergoing recombination will be analyzed using the general settings shown in the fifth column of Table 1. As will next be shown, the kind of evolutionary dynamics exhibited by these populations is, nonetheless, of the same kind as the homogenizing dynamics characteristic of populations undergoing only one type of recombination at a time.

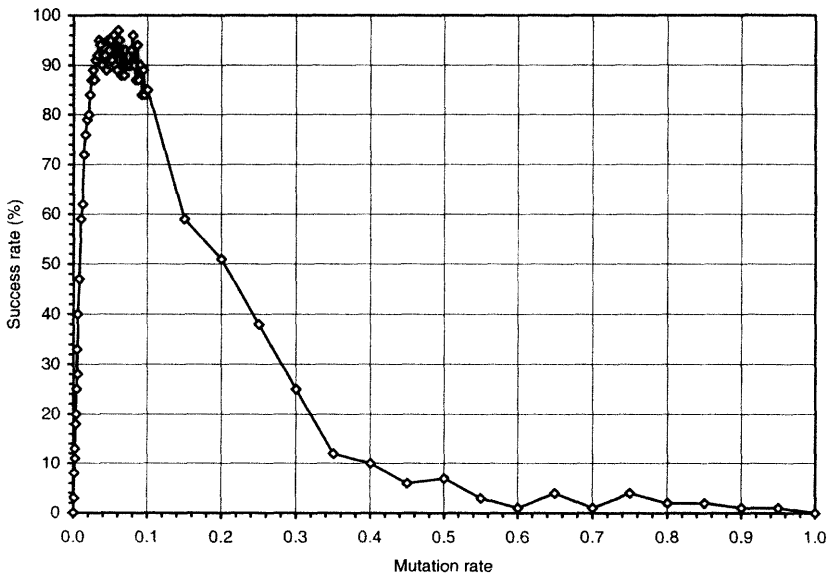


Figure 1. Determining the performance summit using mutation alone.

Table 1. Success rates and parameters for a non-homogenizing system undergoing mutation (Mut) and homogenizing systems undergoing two-point recombination (Rec2P), one-point recombination (Rec1P), gene recombination (RecG), and three different kinds of recombination (RecMix).

	Mut	Rec2P	Rec1P	RecG	RecMix
Number of runs	100	100	100	100	100
Number of generations	100	100	100	100	100
Population size	50	50	50	50	50
Number of fitness cases	10	10	10	10	10
Head length	6	6	6	6	6
Number of genes	6	6	6	6	6
Chromosome length	78	78	78	78	78
Mutation rate	0.05	--	--	--	--
Two-point recombination rate	--	1.0	--	--	0.8
One-point recombination rate	--	--	1.0	--	0.8
Gene recombination rate	--	--	--	1.0	0.8
Selection range	20%	20%	20%	20%	20%
Precision	0%	0%	0%	0%	0%
Success rate	96%	0.04%	0.03%	0.0%	0.13%

3.2 Choosing Non-homogenizing and Homogenizing Populations to Study the Founder Effect

In GEP, populations undergoing mutation are characterized by non-homogenizing dynamics where a considerable gap between average and best fitness is maintained throughout the evolutionary history of a population [5]. Furthermore, in this kind of dynamics, the plot for average fitness shows a pronounced oscillatory pattern which reveals the extent of the modifications taking place in the genome of the individuals. Figure 2 shows such a dynamics obtained for a successful run of the experiment summarized in the first column of Table 1.

On the other hand, populations undergoing recombination alone have homogenizing dynamics [5]. In these populations, the gap between average and best fitness is considerably smaller and, with time, tends to disappear completely. Obviously, when this happens all the individuals in the population have the same genetic makeup and populations become stagnant and incapable of adaptation. Also important is the fact that the plot for average fitness does

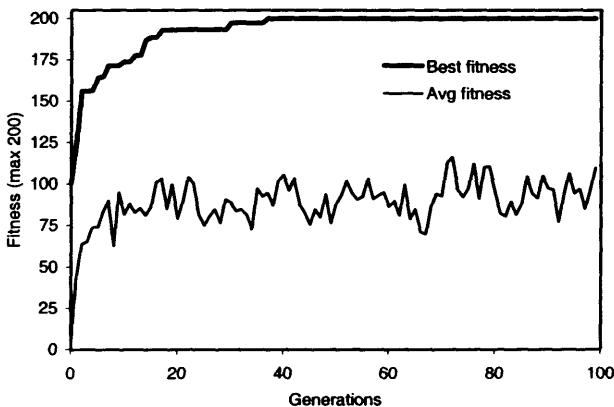


Figure 2. Evolutionary dynamics characteristic of non-homogenizing systems. In this case, the population evolved under a mutation rate of 0.05. Note the oscillatory pattern on average fitness and the wide gap between best and average fitness.

not show such dramatic oscillations as observed in populations with non-homogenizing dynamics. Note also that, in these systems, populations evolve very inefficiently (see Table 1).

The evolutionary dynamics presented in Figure 3 was obtained for populations subjected to three different kinds of recombination simultaneously (Table 1, column 5). Notwithstanding, these populations exhibit the same homogenizing effect described for populations undergoing only one type of recombination at a time. This further reinforces the hypothesis that recombination is conservative and, therefore, plays a major role at maintaining the status quo [5]. Note that, in this particular case, by generation 54 the plot for average fitness meets the plot for best fitness and all individuals become genetically identical. This might be seen as a good thing especially if all the individuals would have become equal and perfect. Recall, however, that in complex real-world problems, as in nature, perfection is always a step further ahead. The disadvantages of such an evolutionary strategy, however, become evident when average fitness reaches best fitness before a perfect or good solution is found. Figure 4 shows such a case where the population stabilized on a mediocre solution. In this case, after generation 86 adaptation becomes impossible because all individuals share the same genetic makeup. Indeed, the small success rates typical of populations undergoing recombination alone (see Table 1, for instance) indicate that, most of the times, homogenizing populations converge before finding a good solution because they became irrevocably stuck in some local point, not necessarily optimal.

It is worth noticing that in the experiments summarized in Table 1, totally random initial populations were used and, therefore, the number of viable individuals in those initial populations was not controlled. In the next section it is shown how the number of viable individuals in initial populations can be rigorously controlled in order to analyze the founder effect in artificial evolutionary systems.

4. Analyzing the Founder Effect in Simulated Evolutionary Processes

The question of the initial diversity is one of interest in artificial evolutionary systems as a considerable amount of computational resources could go into guaranteeing the adequate initial diversity for populations to evolve efficiently. Here, two different systems will be com-

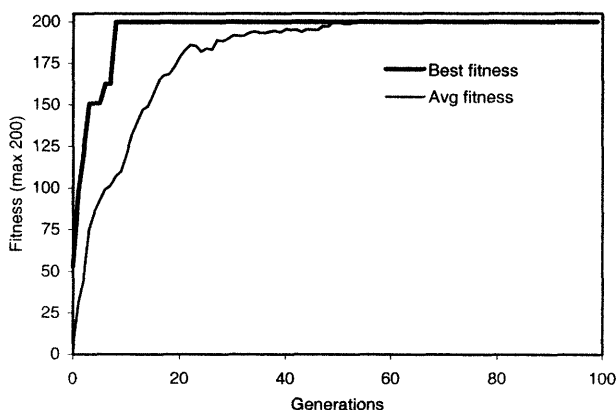


Figure 3. Evolutionary dynamics characteristic of homogenizing populations undergoing recombination. The rates of the three recombination operators used (two-point, one-point and gene recombination) were identical and equal to 0.8. Note the absence of dramatic oscillations on average fitness and that average fitness increases consistently until the complete loss of genetic diversity.

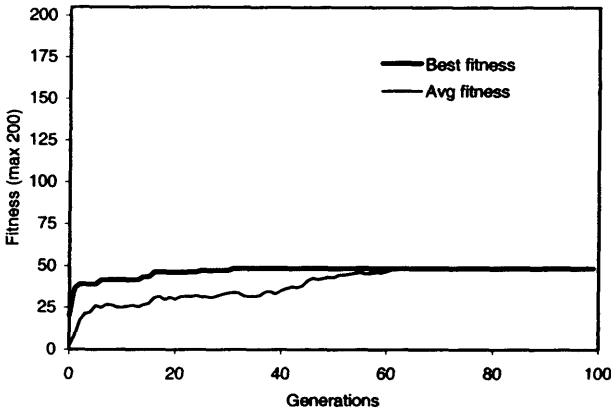


Figure 4. Convergence on a mediocre solution in homogenizing populations undergoing recombination alone. The parameters are exactly the same as in Figure 3.

pared: one that relies on mutation and has a non-homogenizing evolutionary dynamics and another that relies exclusively on crossover and, therefore, has a homogenizing dynamics.

For this analysis, a smaller, totally random “initial” population (founder population) composed of a certain number of viable individuals is created. That is, the run only starts when all the members of the founder population are viable, that is, have positive fitness. These founder individuals are afterwards selected and reproduced, leaving as many descendants as the actual population size P .

As shown in Figure 5, for non-homogenizing populations there is no correlation between success rate and the initial diversity. Indeed, due to the constant introduction of genetic modification in the population, in non-homogenizing populations, after a certain time, the founder effect is completely erased and populations evolve, as usual, efficiently.

However, a very different situation happens in populations where crossover is the only source of genetic diversity and the evolutionary dynamics are homogenizing in effect. In these cases, there is a strong correlation between success rate and initial diversity. Note that

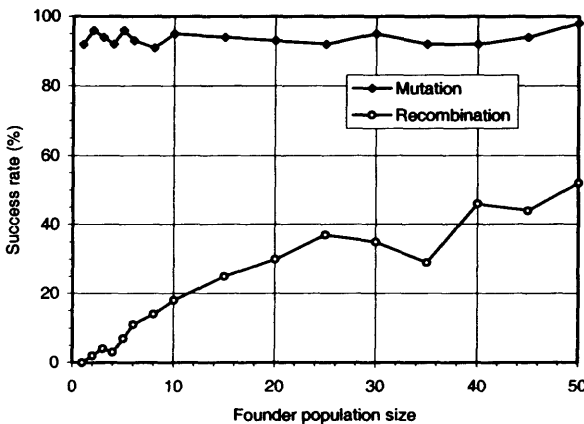


Figure 5. Dependence of success rate on the size of the founder population in non-homogenizing populations undergoing mutation alone (mutation rate equal to 0.05) and homogenizing populations undergoing recombination alone (two-point, one-point and gene recombination rates all equal to 0.8).

populations evolve poorly under recombination, being practically incapable of adaptation in the cases where only 2-5 founder individuals are used (obviously, for cases with only one founder, homogenizing populations are altogether incapable of adaptation). It is worth emphasizing that, in these systems, even when the size of the founder population is equal to P , the success rate is significantly smaller than in populations undergoing mutation, with only one viable individual in the founder population.

Also worth considering is that the computational resources required to guarantee the creation of large founder populations are very expensive. Thus, systems such as GEP capable of evolving efficiently with minimal initial diversity are most advantageous. Furthermore, for some complex problems like, for instance, the discovery of cellular automata rules for the density-classification task [4], it is very difficult to generate randomly a viable individual, even a mediocre one, to start the run. In those cases, systems like GEP can use this individual as founder and continue from there, whereas systems relying on recombination alone will be stuck for a long time before they gather momentum. In GEP, due to the varied set of genetic operators available such as the high-performing point mutation and transposition, there is no need for large founder populations because as long as one viable individual is randomly generated in the initial population the evolutionary process can get started.

5. Conclusions

The question of the initial diversity in artificial evolutionary systems was addressed using gene expression programming. Due to the varied set of genetic operators and the high efficiency of the algorithm, it was possible to compare dissimilarly performing systems such as systems evolving under mutation alone and systems undergoing only recombination. As most existing artificial evolutionary systems rely either on mutation or recombination, this analysis can help understand the different evolutionary strategies followed by each system.

The results obtained in this work show that, on the one hand, systems using the high-performing mutation operator are not only more efficient but also capable of adaptation under extreme evolutionary bottlenecks. In fact, these systems show no correlation between the size of the founder population and success rate. Consequently, in the course of a run, this kind of system is never caught in evolutionary cul-de-sacs and, therefore, evolves without end.

On the other hand, systems relying on recombination alone not only perform poorly but also are unable to adapt and evolve when populations pass through a really tight bottleneck. Consequently, these systems not only are useless whenever only one viable individual is available to start an evolutionary epoch but also frequently become irrevocably stuck at evolutionary cul-de-sacs the system itself creates. Because of this, it is mandatory that these systems guarantee a high level of genetic diversity in initial populations for one thing, and for another, the population sizes in these systems must be huge in order to prevent evolutionary cul-de-sacs from happening. Obviously, these systems are highly expensive and impractical.

References

- [1] Banzhaf, W., Genotype-Phenotype-Mapping and Neutral Variation – A Case Study in Genetic Programming. In Y. Davidor, H.-P. Schwefel, and R. Männer, eds., *Parallel Problem Solving from Nature III, Lecture Notes in Computer Science*, 866: 322-332, Springer-Verlag, 1994.
- [2] Cramer, N. L., A Representation for the Adaptive Generation of Simple Sequential Programs. In J. J. Grefenstette, ed., *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, 183-187, Erlbaum, 1985.

- [3] Dawkins, R., *River out of Eden*. Weidenfeld and Nicolson, 1995.
- [4] Ferreira, C., 2001. Gene Expression Programming: A New Adaptive Algorithm for Solving Problems. *Complex Systems*, 13 (2): 87-129.
- [5] Ferreira, C., Mutation, Transposition, and Recombination: An Analysis of the Evolutionary Dynamics. In H. J. Caulfield, S.-H. Chen, H.-D. Cheng, R. Duro, V. Honavar, E. E. Kerre, M. Lu, M. G. Romy, T. K. Shih, D. Ventura, P. P. Wang, Y. Yang, eds., *Proceedings of the 6th Joint Conference on Information Sciences, 4th International Workshop on Frontiers in Evolutionary Algorithms*, 614-617, Research Triangle Park, North Carolina, USA, 2002.
- [6] Fogel, D. B. and J. W. Atmar, 1990. Comparing Genetic Operators with Gaussian Mutations in Simulated Evolutionary Processes Using Linear Systems. *Biological Cybernetics* 63: 111-114.
- [7] Fogel, L. J., A. J. Owens, and M. J. Walsh, *Artificial Intelligence Through Simulated Evolution*, New York, Wiley Publishing, 1966.
- [8] Holland, J. H., *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press, 1975 (second edition: MIT Press, 1992).
- [9] Koza, J. R., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, Cambridge, MA: MIT Press, 1992.
- [10] Mayr, E., Change of Genetic Environment and Evolution. In J. Huxley, A. C. Hardy, and E. B. Ford, eds., *Evolution as a Process*, 157-180, Allen and Unwin, London, 1954.
- [11] Mayr, E., *Animal Species and Evolution*, Harvard University Press, Cambridge, Massachusetts, 1963.
- [12] Rechenberg, I., *Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*, Frommann-Holzboog, Stuttgart, 1973.

Hybrid Evolutionary Multi-Objective Optimization Algorithms

Hisao ISHIBUCHI and Tadashi YOSHIDA

*Department of Industrial Engineering, Osaka Prefecture University
1-1 Gakuen-cho, Sakai, Osaka 599-8531, Japan*

Abstract. This paper examines how the search ability of evolutionary multi-objective optimization (EMO) algorithms can be improved by the hybridization with local search through computational experiments on multi-objective permutation flowshop scheduling problems. The task of EMO algorithms is to find a variety of non-dominated solutions of multi-objective optimization problems. First we describe our multi-objective genetic local search (MOGLS) algorithm, which is the hybridization of a simple EMO algorithm with local search. Next we discuss some implementation issues of local search in our MOGLS algorithm such as the choice of initial (i.e., starting) solutions for local search and a termination condition of local search. Then we implement hybrid EMO algorithms using well-known EMO algorithms: SPEA and NSGA-II. Finally we compare those EMO algorithms with their hybrid versions through computational experiments. Experimental results show that the hybridization with local search can improve the search ability of the EMO algorithms when local search is appropriately implemented in their hybrid versions.

1. Introduction

Since Schaffer's work [1], evolutionary algorithms have been applied to various multi-objective optimization problems for finding their Pareto-optimal solutions. Evolutionary algorithms for multi-objective optimization are often referred to as EMO (evolutionary multi-objective optimization) algorithms. For review of this field, see [2], [3]. The task of EMO algorithms is to find Pareto-optimal solutions as many as possible. It is, however, impractical to try to find true Pareto-optimal solutions of large-scale combinatorial optimization problems. Thus non-dominated solutions among examined ones are presented to decision makers as a result of the search by EMO algorithms. In this case, EMO algorithms try to drive populations to true Pareto-optimal solutions as close as possible for obtaining a variety of near Pareto-optimal solutions.

One promising approach for improving the search ability of EMO algorithms to find near Pareto-optimal solutions is the hybridization with local search. The hybridization of evolutionary algorithms with local search has already been investigated in many studies for single-objective optimization problems [4], [5]. Such a hybrid algorithm is often referred to as memetic algorithms. A hybrid evolutionary algorithm with local search for multi-objective optimization was first implemented in [6], [7] as a multi-objective genetic local

search (MOGLS) algorithm. Jaskiewicz [8] improved the performance of the MOGLS algorithm by modifying the selection mechanism for choosing parent solutions for crossover in its EMO part. The performance of the MOGLS algorithm was also improved by introducing a selection mechanism into its local search part for choosing good solutions to which local search is applied in each generation [9], [10]. Knowles & Corne [11] combined their Pareto archived evolution strategy (PAES [12]) with a crossover operation for designing a memetic PAES (M-PAES). In their M-PAES, the Pareto-dominance relation and the grid-type partition of the multi-dimensional objective space were used for determining the acceptance (or rejection) of new solutions generated in genetic search and local search. The M-PAES had a special form of elitism inherent in the PAES.

Generic frameworks of hybrid EMO algorithms are shown in Fig. 1. The two frameworks in Fig. 1 are the same except for the order of genetic search and local search. Genetic operations are first applied to an initial population in Fig. 1 (a). On the other hand, genetic operations are applied after an initial population is improved by local search in Fig. 1 (b). The two frameworks are executed in the same manner after the second generation: the EMO part and the local search part are iterated for finding Pareto-optimal solutions. Emphasis is implicitly placed on the local search part in Fig. 1 (b) while the EMO part is implicitly viewed as the main part in Fig. 1 (a). In this paper, we use the framework in Fig. 1 (a) for describing hybrid EMO algorithms. In Fig. 1 (a), the local search part can be also viewed as a special kind of mutation in EMO algorithms.

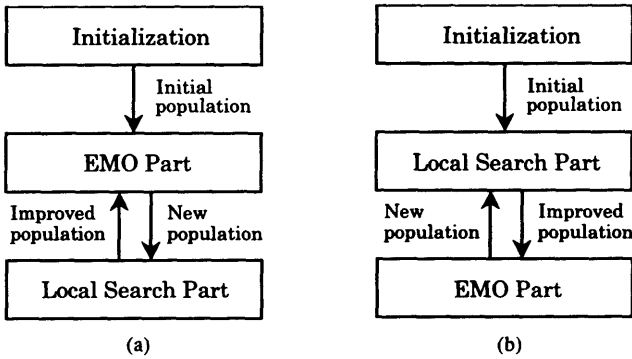


Fig. 1. Two generic frameworks of hybrid EMO algorithms. In (a), local search is applied to new solutions generated by the EMO part. In (b), genetic operations are applied to an improved population generated by the local search part.

In this paper, we first rewrite the MOGLS algorithm in our former studies [9], [10] by introducing a new parameter: local search probability P_{LS} . Local search is probabilistically applied to each solution with the local search probability P_{LS} in our MOGLS algorithm. Next we discuss some implementation issues of local search in our MOGLS algorithm such as the choice of initial (i.e., starting) solutions for local search and a termination condition of local search. In the local search part of our MOGLS algorithm, good solutions are selected from the current population as initial solutions for local search. Then local search is probabilistically applied to each of the selected solutions. For decreasing the CPU time spent by local search (i.e., for striking a balance between genetic search and local search

[10]), we use an early termination strategy where local search is terminated before finding a locally optimal solution. The probabilistic application of local search and its early termination are for preventing a possible negative effect of local search: It may cause a premature convergence to local solutions. Then we implement hybrid versions of well-known EMO algorithms: SPEA (strength Pareto evolutionary algorithm [13]) and NSGA-II (revised non-dominated sorting genetic algorithm [14]). The SPEA and the NSGA-II are combined with local search in the framework shown in Fig. 1 (a). Finally we compare those EMO algorithms with their hybrid versions through computational experiments on multi-objective permutation flowshop scheduling problems.

2. Multi-Objective Genetic Local Search Algorithm

Let us consider the following n -objective minimization problem:

$$\text{Minimize } \mathbf{z} = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_n(\mathbf{x})), \quad (1)$$

$$\text{subject to } \mathbf{x} \in \mathbf{X}, \quad (2)$$

where \mathbf{z} is the objective vector, $f_i(\mathbf{x})$ is the i -th objective to be minimized, \mathbf{x} is the decision vector, and \mathbf{X} is the feasible region in the decision space.

The EMO part of our MOGLS algorithm is a simple multi-objective genetic algorithm with the roulette wheel selection based on a scalar fitness function. It has a secondary population for elitism [15]. As in standard single-objective genetic algorithms, first an initial population of N_{pop} solutions is randomly generated where N_{pop} is the population size. Non-dominated solutions in the initial population are identified, and a secondary population is constructed from their copies. The fitness value of each solution in the current population is evaluated using the following scalar fitness function (to be minimized):

$$f(\mathbf{x}) = w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) + \dots + w_n f_n(\mathbf{x}), \quad (3)$$

where w_i is a normalized non-negative random weight for the i -th objective function $f_i(\mathbf{x})$: $w_1 + w_2 + \dots + w_n = 1$ and $w_i \geq 0$ for $\forall i$. A pair of parent solutions is selected from the current population using the roulette wheel selection scheme with the linear scaling:

$$P_S(\mathbf{x}) = \frac{f_{\max}(\Psi) - f(\mathbf{x})}{\sum_{\mathbf{y} \in \Psi} \{f_{\max}(\Psi) - f(\mathbf{y})\}}, \quad (4)$$

where $P_S(\mathbf{x})$ is the selection probability of the solution \mathbf{x} , Ψ denotes the current population, and $f_{\max}(\Psi)$ is the largest (i.e., worst) fitness value $f(\mathbf{x})$ in the current population Ψ under the current weight vector $\mathbf{w} = (w_1, w_2, \dots, w_n)$. For finding a variety of non-dominated solutions (i.e., for realizing various search directions in the n -dimensional objective space), the weight vector is randomly specified whenever a pair of parent solutions is to be selected. That is, the selection of each pair of parent solutions is governed by a different weight vector. A pre-specified number of pairs of parent solutions are selected in this manner. New solutions are generated by crossover and mutation in the same manner as standard single-objective genetic algorithms. Then a pre-specified number of non-dominated solutions are randomly selected from the secondary population as elite solutions, and their copies are added to the newly generated solutions. In this manner, a new population is constructed from the solutions generated by the genetic operations and the copies of non-dominated solutions randomly selected from the secondary population.

Next the local search part receives the new population generated in the EMO part. In our MOGLS algorithm, an initial (i.e., starting) solution for local search is selected from the current population (i.e., population generated in the EMO part) using the scalar fitness function in (3) with random weight values. Weight values are randomly specified whenever an initial solution is to be selected for local search. That is, each initial solution is selected based on a different weight vector. Local search for the selected initial solution is governed by the scalar fitness function with the weight vector used in its selection. The tournament selection with the tournament size five is used in our computational experiments for the selection of an initial solution for local search. Local search is probabilistically applied to a copy of the selected initial solution with the local search probability P_{LS} . When local search is not applied, a copy of the selected initial solution is added to the next population. When local search is applied, a neighbor of the current solution is randomly selected. If the neighbor is superior to the current solution with respect to the scalar fitness function with the current weight vector, the current solution is immediately replaced with the neighbor. That is, we use the hill-climbing algorithm with the first improvement strategy in the local search part instead of the best improvement (i.e., steepest hill-climbing) strategy. For preventing the local search part from spending almost all the available CPU time, we restrict the number of successive fails of local move to k where k is a user-definable parameter. This means that local search for the current solution is terminated when a better solution is not found among k neighbors randomly selected from the neighborhood of the current solution. Of course, when a better neighbor is found, we can examine k neighbors of the new current solution again. When local search is terminated, the current solution is added to the new population. The selection of an initial solution from the current population and the probabilistic application of local search are iterated N_{pop} times for generating the new population with N_{pop} solutions (i.e., for improving the population generated by the EMO part).

The outline of our MOGLS algorithm can be written as follows:

MOGLS Algorithm

Step 0) Initialization: Randomly generate an initial population of N_{pop} solutions.

[EMO Part]

Step 1) Evaluation: Calculate the n objectives for each solution in the current population. Then update the secondary population where non-dominated solutions are stored separately from the current population.

Step 2) Selection: Repeat the following procedures to select $(N_{pop} - N_{elite})$ pairs of parent solutions where N_{elite} is the number of elite solutions.

(a) Randomly specify the weight values w_1, w_2, \dots, w_n .

(b) Select a pair of parent solutions using the scalar fitness function in (3). The roulette wheel scheme in (4) is used for the selection of parent solutions.

Step 3) Crossover and mutation: Apply a crossover operation to each of the selected $(N_{pop} - N_{elite})$ pairs of parent solutions. A new solution is generated from each pair. Then apply a mutation operation to each of the generated new solutions.

Step 4) Elitist strategy: Randomly select N_{elite} solutions from the secondary population. Then add their copies to the $(N_{pop} - N_{elite})$ solutions generated in Step 3 to construct a population of N_{pop} solutions.

[Local Search Part]

Step 5) Local search: Iterate the following three steps N_{pop} times. Then replace the

current population with N_{pop} solutions obtained by the following steps.

- (a) Randomly specify the weight values w_1, w_2, \dots, w_n .
- (b) Select a solution from the current population using tournament selection with replacement based on the scalar fitness function in (3) with the current weight vector specified in (a). A copy of the selected solution is used in (c). Thus no solution is removed from the current population.
- (c) Apply local search to a copy of the selected solution using the current weight vector with the local search probability P_{LS} . When local search is applied, the current solution after local search is included in the next population. As mentioned above, local search is terminated when no better solution is found among k neighbors randomly generated from the current solution. On the other hand, when local search is not applied, a copy of the selected solution in (b) is added to the next population.

Step 6) Return to Step 1.

This algorithm is terminated when a pre-specified stopping condition is satisfied. In this paper, we use the number of examined solutions as the stopping condition for comparing different algorithms under the same computation load.

Let us demonstrate how the performance of the EMO algorithm can be improved by the hybridization with local search in our MOGLS algorithm. As test problems, we generated eight flowshop scheduling problems in the same manner as [7]. The processing time of each job on each machine was specified as a random integer in the interval $[1, 99]$. The due date of each job was specified by adding a random integer in the interval $[-100, 100]$ to its actual completion time in a randomly generated schedule. All the eight test problems have 20 machines. Using the number of objectives (n) and the number of jobs (N), we denote each of the eight test problems as n/N where $n = 2, 3$ and $N = 20, 40, 60, 80$. Four test problems have two objectives (i.e., $n = 2$): to minimize the makespan and to minimize the maximum tardiness. The other four test problems are three-objective problems (i.e., $n = 3$) with an additional objective: to minimize the total flow time.

In our computational experiments of this paper, we used the same genetic operations as [7]: the two-point order crossover and the shift change mutation. The shift change mutation is the same as the insertion operation: remove a randomly selected job and insert it into another position. The shift change mutation was also used in the local search part for generating a neighbor from the current solution as in [7]. We used the following parameter specifications in the EMO part:

Population size (N_{pop}): 60,
 Crossover probability: 0.9,
 Mutation probability per string: 0.6,
 Number of elite solutions (N_{elite}): 10,
 Stopping condition: Evaluation of 100,000 solutions.

Several different specifications were examined for the two parameters P_{LS} and k .

A solution set obtained by our MOGLS algorithm was evaluated by the average normalized distance from each reference solution (i.e., approximate Pareto-optimal solution) to the nearest solution in the obtained solution set. This performance criterion can measure the proximity of the obtained solution set to the Pareto front and the quality of the distribution of solutions in the obtained solution set. The reference solutions were found from ten independent runs of the MOGLS algorithm, the SPEA, and the NSGA-II for each

test problem with much longer CPU time (i.e., five million solutions were examined in each run of each algorithm). We compared 30 solution sets, which were obtained from ten runs of the three algorithms, with each other for finding non-dominated solutions. All the non-dominated solutions were used as reference solutions for each test problem. The objective space of each test problem was normalized so that the minimum and maximum values of each objective among the reference solutions became 0 and 100, respectively.

Using the reference solutions of each test problem with the normalized objective space, we evaluated solution sets obtained by the MOGLS algorithm with various specifications of the two parameters in the local search part: P_{LS} and k . We also evaluated the performance of the EMO part of the MOGLS algorithm. Average results over 50 independent runs are summarized in Table 1. From this table, we can see that the performance of the simple EMO algorithm was significantly improved by the hybridization with local search. We can also see that the performance of our MOGLS algorithm strongly depended on the two parameters: k and P_{LS} . The best result for each test problem in each table is indicated by “*”. The best result for each test problem among all tables in this paper is underlined.

The performance of the MOGLS algorithm is significantly deteriorated when we remove the selection mechanism (i.e., selection of initial solutions for local search) from the local search part. We performed the same computational experiments as Table 1 using the MOGLS algorithm without the selection mechanism in the local search part. In this case, local search was probabilistically applied to every solution in the current population with the local search probability P_{LS} independent of their fitness values. When local search was applied to a solution \mathbf{x} , the local search direction for \mathbf{x} (i.e., weight vector in the scalar fitness function) was specified using the concept of pseudo-weight vector [2]. The pseudo-weight w_i for the i -th objective $f_i(\mathbf{x})$ was defined for the solution \mathbf{x} as

$$w_i = \frac{f_i^{\max} - f_i(\mathbf{x})}{f_i^{\max} - f_i^{\min}} \bigg/ \sum_{j=1}^n \frac{f_j^{\max} - f_j(\mathbf{x})}{f_j^{\max} - f_j^{\min}}, \quad i = 1, 2, \dots, n, \quad (5)$$

where f_i^{\max} and f_i^{\min} are the maximum and minimum values of the i -th objective $f_i(\mathbf{x})$ in the current population, respectively. It should be noted that the local search direction for each initial solution was specified by the weight vector used in the selection of that initial solution in the MOGLS algorithm with the selection mechanism in Table 1. Table 2 shows average results over 50 independent runs of the MOGLS algorithm without the selection mechanism. From the comparison between Table 1 and Table 2, we can see that the performance of the MOGLS algorithm was significantly deteriorated in Table 2 by removing the selection mechanism. That is, we can see that the selection of initial solutions for local search plays a significant role in the MOGLS algorithm. When the local search probability is very small (e.g., $P_{LS} = 0.01$), the selection of initial solutions for local search can be viewed as the selection of good solutions from the current population for constructing the next population. The effect of such selection may be twofold: the improvement in the convergence speed to Pareto-optimal solutions and the decrease in the variety of solutions. Since our MOGLS algorithm uses a very simple EMO algorithm with the parent selection scheme based on the roulette wheel, the positive effect (i.e., the improvement in the convergence speed) dominates the negative effect (i.e., the decrease in the variety of solutions) in our computational experiments. As a result, the selection of initial solutions for local search significantly improved the performance of the MOGLS algorithm from Table 2 to Table 1 even when the local search probability was very small.

Table 1. Performance of the MOGLS algorithm with the selection mechanism in the local search part.

Test Problem	$P_{LS} = 0.01$			$P_{LS} = 0.05$			$P_{LS} = 0.1$			$P_{LS} = 1$			EMO Part
	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	
2/20	5.85	5.59	4.92*	6.14	5.67	4.94	6.23	5.97	4.93	5.89	5.75	5.39	42.61
2/40	22.11	20.65	18.68	21.17	19.27	19.01	20.15	18.50	19.43	18.01*	20.27	26.26	109.76
2/60	24.46	24.08	20.56	24.10	22.16	20.02*	23.64	21.88	20.89	22.55	23.41	27.69	107.52
2/80	110.28	89.50	70.67*	96.73	84.41	71.72	97.84	74.52	74.42	83.18	89.79	127.13	610.29
3/20	8.99	9.13	7.96*	8.96	8.66	8.25	9.05	8.44	8.42	9.45	9.14	9.50	50.06
3/40	22.07	21.15	20.19*	21.67	21.70	21.30	21.70	21.56	20.86	21.25	22.92	26.51	109.91
3/60	32.19	30.72	26.84	31.96	29.29	26.82*	30.35	29.00	27.18	30.52	32.23	35.91	130.88
3/80	36.93	35.00	33.18	35.98	34.93	33.93	34.17	34.91	32.84*	35.83	38.90	47.49	173.60

Table 2. Performance of the MOGLS algorithm without the selection mechanism in the local search part.

Test Problem	$P_{LS} = 0.01$			$P_{LS} = 0.05$			$P_{LS} = 0.1$			$P_{LS} = 1$			EMO Part
	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	
2/20	38.94	22.80	9.89	29.37	19.91	9.21	26.39	19.10	9.29	22.65	12.80	7.33*	42.61
2/40	100.28	66.53	39.13	80.56	56.47	37.64	67.54	53.76	37.70	47.41	43.64	34.69*	109.76
2/60	95.33	61.87	35.52	76.88	52.22	34.41	67.90	50.99	33.81*	46.30	41.09	33.97	107.52
2/80	565.8	390.5	181.3	479.4	338.0	170.3	428.9	305.9	161.6*	277.3	247.7	185.9	610.29
3/20	45.57	23.25	12.68	34.88	19.53	11.80	27.47	18.43	12.04	17.92	13.38	10.48*	50.06
3/40	101.15	60.13	32.68*	78.58	52.59	33.08	68.16	49.84	33.50	42.70	40.74	33.20	109.91
3/60	119.40	73.24	43.84	97.68	65.01	42.29	82.92	62.56	41.95	58.32	53.36	41.12*	130.88
3/80	161.62	97.92	54.98*	131.53	86.58	55.03	114.66	82.98	55.03	74.79	70.15	59.42	173.60

3. Hybrid SPEA and Hybrid NSGA-II

Since the local search part (i.e., Step 5 of our MOGLS algorithm) is independent of the EMO part, it can be combined with other EMO algorithms. In the hybridization with local search, we do not have to modify EMO algorithms. When EMO algorithms have a secondary population, it is updated after the current population is improved by local search. We implemented a hybrid SPEA and a hybrid NSGA-II by replacing the EMO part of the MOGLS algorithm with the SPEA and the NSGA-II, respectively. The hybridization was implemented in the framework of Fig. 1 (a). In those hybrid EMO algorithms, local search was applied to solutions in the primary population (i.e., it was not applied to the secondary population) as in the MOGLS algorithm.

In the same manner as Table 1, we applied the hybrid SPEA and the hybrid NSGA-II to the eight test problems. Average results over 50 independent runs are summarized in Table 3 and Table 4. In the computational experiments in Table 3 and Table 4, the selection mechanism was used in the local search part as in Table 1. In these tables, boldface shows that better results were obtained by the hybrid versions than the non-hybrid EMO algorithms. From Table 3, we can see that the performance of the SPEA was improved by the hybridization with local search for the two-objective test problems. On the other hand, we can see from Table 4 that the performance of the NSGA-II was improved by the hybridization for all test problems when the parameter specification was appropriate (e.g., $P_{LS} = 0.1$ and $k = 100$). It is very interesting to observe that the hybrid SPEA in Table 3 and the hybrid NSGA-II in Table 4 do not always outperform the MOGLS algorithm while their EMO parts clearly outperform the EMO part of our MOGLS algorithm in Table 1.

Table 3. Performance of the hybrid SPEA with the selection mechanism in the local search part.

Test Problem	$P_L = 0.01$			$P_L = 0.05$			$P_L = 0.1$			$P_L = 1$			SPEA
	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	
2/20	7.16	7.27	5.22*	6.65	6.73	5.39	7.19	6.72	5.40	6.69	6.46	5.49	6.05
2/40	17.24	17.60	14.94*	17.14	17.01	16.67	17.11	16.93	18.24	16.83	19.01	26.64	18.01
2/60	23.63	22.47	21.17*	23.00	23.42	21.59	22.33	23.19	21.27	23.03	24.93	27.45	22.09
2/80	80.02	76.38	52.84*	80.38	71.04	54.78	76.22	66.85	62.99	76.92	72.87	125.99	81.18
3/20	8.16	8.01	7.96	8.23	8.40	8.14	7.98	8.42	8.23	8.35	8.62	8.90	7.02*
3/40	19.17	18.26	19.10	18.90	19.42	19.53	18.74	19.70	19.79	19.41	21.60	25.03	15.81*
3/60	25.79	24.72	24.32	25.14	24.93	25.45	25.19	26.37	26.35	26.46	30.05	34.55	24.10*
3/80	32.21	31.91	30.85	31.91	33.12	31.63	31.81	32.99	32.25	33.64	37.02	46.57	28.10*

Table 4. Performance of the hybrid NSGA-II with the selection mechanism in the local search part.

Test Problem	$P_L = 0.01$			$P_L = 0.05$			$P_L = 0.1$			$P_L = 1$			NSGA-II
	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	
2/20	17.21	15.34	13.88	17.90	15.17	9.90	17.29	13.61	8.33	14.52	9.53	5.60*	9.25
2/40	38.84	38.94	35.72	37.66	36.20	24.65	37.24	34.54	19.83*	33.18	21.23	27.44	21.54
2/60	28.02	26.70	25.05	27.30	25.05	21.60	26.65	25.10	21.24*	25.80	24.15	28.47	22.04
2/80	101.69	91.51	81.67	90.49	81.24	70.19	101.34	82.83	68.78*	85.93	81.34	128.42	78.43
3/20	26.26	26.51	23.42	25.89	24.33	17.82	26.74	23.62	16.01	23.82	15.10	9.63*	21.12
3/40	44.48	44.56	34.57	44.94	40.14	28.79	44.24	38.83	25.50*	36.96	27.62	26.22	43.26
3/60	50.07	48.04	36.57	48.89	44.87	31.46	49.47	41.61	29.25*	40.78	33.29	35.48	46.35
3/80	49.32	46.28	39.98	47.92	43.37	35.25	46.59	41.64	33.68*	41.80	38.41	48.13	44.59

While the performance of the EMO part in the MOGLS algorithm was significantly improved by the hybridization, the improvement was not large in Table 3 and Table 4. One possible cause of the limited improvement is the negative effect of the selection mechanism in the local search part (i.e., the decrease in the variety of solutions). Thus we examined the performance of the hybrid SPEA and NSGA-II with no selection mechanism in the local search part as in Table 2. Average results are summarized in Table 5 and Table 6. From the comparison of Table 5 with Table 3, we can see that the performance of the hybrid SPEA for the three-objective problems was improved by removing the selection mechanism while the performance for the two-objective problems was deteriorated. This may be because the variety of solutions is more important in the three-objective problems with much more Pareto-optimal solutions. On the other hand, the performance of the hybrid NSGA-II was improved for both the two-objective and three-objective problems in many combinations of P_L and k . Since the NSGA-II does not have a secondary population, the negative effect of the decrease in the variety of solutions may be more severe than the SPEA.

In the above computational experiments, we always used the scalar fitness function in the local search part. We also examined the use of the dominance relation based on the n objectives. In this case, a local move to a neighbor was accepted only when the current solution was dominated by the neighbor. As in Table 5 and Table 6 (i.e., without the selection mechanism in the local search part), we examined the performance of the hybrid SPEA and the hybrid NSGA-II with the acceptance criterion of a local move based on the dominance relation. Average results are summarized in Table 7 and Table 8. From Table 5 ~ Table 8, we can see that the use of the dominance relation as the acceptance criterion did not improve the performance of the hybrid algorithms in our computational experiments.

Table 5. Performance of the hybrid SPEA without the selection mechanism in the local search part.

Test Problem	$P_L = 0.01$			$P_L = 0.05$			$P_L = 0.1$			$P_L = 1$			SPEA
	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	
2/20	6.21	5.93*	5.99	6.13	6.48	6.20	6.45	6.59	6.28	6.72	7.59	6.32	6.05
2/40	17.13*	18.39	19.50	18.77	18.64	22.13	18.78	20.35	23.45	19.82	28.74	30.09	18.01
2/60	22.56	21.26*	21.75	23.02	22.67	23.70	22.40	24.18	23.96	24.75	29.96	29.56	22.09
2/80	81.04	76.07*	76.41	76.12	76.29	92.11	79.39	81.12	102.28	87.43	125.00	153.24	81.18
3/20	6.93*	7.09	7.19	6.98	7.23	8.03	7.18	7.21	8.43	7.33	8.52	9.51	7.02
3/40	15.86	16.72	20.48	16.54	17.80	22.82	16.19	18.76	23.84	18.90	28.24	29.02	15.81*
3/60	22.70*	23.23	26.32	23.26	24.42	29.50	24.17	26.77	30.26	26.98	38.94	38.92	24.10
3/80	27.15	26.72*	32.12	28.20	28.71	36.09	28.44	30.82	38.86	31.79	46.65	53.20	28.10

Table 6. Performance of the hybrid NSGA-II without the selection mechanism in the local search part.

Test Problem	$P_L = 0.01$			$P_L = 0.05$			$P_L = 0.1$			$P_L = 1$			NSGA-II
	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	
2/20	9.37	9.04	8.67	8.63	9.18	7.36	8.80	9.13	7.00	8.96	7.80	6.24*	9.25
2/40	22.84	20.81	19.94*	21.48	21.37	21.17	21.72	20.94	21.77	20.77	26.85	30.76	21.54
2/60	21.69	21.88	20.95*	21.55	21.46	22.26	21.93	22.39	23.10	21.74	27.62	30.76	22.04
2/80	77.69	75.40	74.58	74.27*	75.16	90.20	83.14	81.51	98.04	85.18	128.85	163.79	78.43
3/20	22.84	21.55	17.03	22.11	19.86	12.20	21.39	17.97	11.48	19.98	11.74	9.44*	21.12
3/40	44.08	39.83	27.83	43.48	36.26	24.88	45.21	34.37	24.69*	36.31	30.85	29.70	43.26
3/60	43.01	42.52	30.44	43.78	37.66	30.03*	44.23	37.56	31.14	39.12	39.71	40.29	46.35
3/80	44.00	42.07	34.61*	43.90	39.59	36.28	43.10	39.76	39.67	39.53	46.91	54.32	44.59

Table 7. Performance of the hybrid SPEA without the selection mechanism in the local search part. The local move to a neighbor was accepted only when the current solution was dominated by the neighbor.

Test Problem	$P_L = 0.01$			$P_L = 0.05$			$P_L = 0.1$			$P_L = 1$			SPEA
	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	
2/20	5.85*	5.93	6.60	6.06	6.07	7.87	6.18	6.95	8.37	7.08	8.60	10.59	6.05
2/40	17.94	18.04	21.39	18.74	19.85	28.64	17.65*	19.82	31.09	21.14	31.33	46.01	18.01
2/60	22.10	23.24	25.24	22.69	24.30	28.14	22.12	24.83	30.32	25.43	31.73	42.11	22.09*
2/80	79.90*	82.41	97.35	82.27	88.03	135.08	80.86	96.36	164.78	96.51	144.53	316.44	81.18
3/20	7.15	7.18	7.93	7.06	7.32	9.97	7.06	7.69	10.99	7.71	10.12	14.39	7.02*
3/40	16.27	17.20	19.87	16.35	18.57	26.82	16.28	18.30	29.10	19.50	28.61	39.31	15.81*
3/60	22.96*	24.73	28.44	23.41	25.12	34.04	23.61	27.40	37.94	28.09	38.87	50.61	24.10
3/80	27.40	27.26*	33.19	27.65	29.45	40.37	28.27	32.33	45.38	32.33	46.12	65.85	28.10

Table 8. Performance of the hybrid NSGA-II without the selection mechanism in the local search part. The local move to a neighbor was accepted only when the current solution was dominated by the neighbor.

Test Problem	$P_L = 0.01$			$P_L = 0.05$			$P_L = 0.1$			$P_L = 1$			NSGA-II
	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	$k = 1$	10	100	
2/20	9.52	9.30	8.64	8.80	8.52	8.45	9.38	9.00	8.17*	8.98	9.16	10.99	9.25
2/40	21.74	21.53	21.18	20.08*	20.77	26.45	21.06	22.45	28.76	21.37	27.90	48.46	21.54
2/60	21.63	21.84	23.28	22.02	22.21	27.53	21.50*	22.41	30.82	23.35	29.59	43.10	22.04
2/80	82.14	82.02	97.92	75.19*	83.71	150.30	82.15	95.09	194.08	92.72	152.54	322.61	78.43
3/20	22.44	21.30	20.11	22.45	20.20	15.71	22.08	19.92	13.86	20.71	14.37	13.73*	21.12
3/40	44.99	42.99	30.49	43.08	37.07	27.56*	42.52	35.37	29.34	35.60	29.99	40.49	43.26
3/60	45.05	42.41	33.75	45.57	40.07	33.71*	44.07	38.00	36.90	39.76	39.90	52.33	46.35
3/80	43.59	44.28	36.89*	44.01	42.13	40.87	44.36	40.95	44.64	41.13	45.96	67.95	44.59

4. Conclusion

In this paper, we examined how the performance of EMO algorithms can be improved by the hybridization with local search. We showed through computational experiments on multi-objective permutation flowshop scheduling problems that the performance of the EMO part of our MOGLS algorithm was improved by the hybridization with local search while the performance of the MOGLS algorithm strongly depended on the choice of the parameter values in the local search part (i.e., k and P_{LS}). We also implemented hybrid versions of the SPEA and the NSGA-II. The hybridization with local search improved the performance of the SPEA and the NSGA-II when k and P_{LS} were appropriately specified. The best results for the two-objective test problems were obtained by our MOGLS algorithm and the hybrid SPEA with the selection mechanism of initial solutions in the local search part. On the other hand, the best results for the three-objective test problems were obtained by the non-hybrid SPEA and the hybrid SPEA with no selection mechanism of initial solutions in the local search part. One potential advantage of hybrid algorithms over pure EMO algorithms is the decrease in the CPU time. This is because local search can be much more efficiently executed than genetic search in many application problems.

This study is partially supported by Japan Society for the Promotion of Science (JSPS) through Grand-in-Aid for Scientific Research (B): KAKENHI (14380194).

References

- [1] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," *Proc. of 1st International Conference on Genetic Algorithms and Their Applications*, pp. 93-100, 1985.
- [2] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, 2001.
- [3] C. A. Coello Coello, D. A. van Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer Academic Publishers, 2002.
- [4] P. Moscato, "Memetic algorithms: A short introduction," in D. Corne, F. Glover, and M. Dorigo (eds.), *New Ideas in Optimization*, McGraw-Hill, pp. 219-234, Maidenhead, 1999.
- [5] Memetic Algorithms' Home Page: http://www.densis.fee.unicamp.br/~moscato/memetic_home.html.
- [6] H. Ishibuchi and T. Murata, "Multi-objective genetic local search algorithm," *Proc. of 3rd IEEE International Conference on Evolutionary Computation*, pp. 119-124, 1996.
- [7] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 28, no. 3, pp. 392-403, 1998.
- [8] A. Jaszkiwicz, "Genetic local search for multi-objective combinatorial optimization," *European Journal of Operational Research*, vol. 137, no. 1, pp. 50-71, 2002.
- [9] H. Ishibuchi, T. Yoshida, and T. Murata, "Selection of initial solutions for local search in multiobjective evolutionary algorithms," *Proc. of 2002 Congress on Evolutionary Computation*, pp. 950-955, 2002.
- [10] H. Ishibuchi and T. Yoshida, and T. Murata, "Balance between genetic search and local search in hybrid evolutionary multi-criterion optimization algorithms," *Proc. of 2002 Genetic and Evolutionary Computation Conference*, pp. 1301-1308, 2002.
- [11] J. D. Knowles and D. W. Corne, "M-PAES: A memetic algorithm for multiobjective optimization," *Proc. of 2000 Congress on Evolutionary Computation*, pp. 325-332, 2000.
- [12] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using Pareto archived evolution strategy," *Evolutionary Computation*, vol. 8, no. 2, pp. 149-172, 2000.
- [13] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, 1999.
- [14] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, 2002.
- [15] T. Murata and H. Ishibuchi, "MOGA: Multi-objective genetic algorithms," *Proc. of 1995 IEEE International Conference on Evolutionary Computation*, pp. 289-294, 1995.

A Permutation Based Genetic Algorithm for RNA Secondary Structure Prediction

Kay C. WIESE and Edward GLEN

Simon Fraser University Surrey, Information Technology

2400 Central City, 10153 King George Highway

Surrey, BC V3T 2W1

Phone: (604) 586.6114, Fax: (604) 586.5237

Email: {wiese@sfu.ca, eglen@sfu.ca}

Abstract

This paper presents a permutation based genetic algorithm (GA) to predict the secondary structure of RNA molecules. More specifically the proposed algorithm predicts which specific canonical base pairs will form hydrogen bonds and build helices, also known as stem loops. Since RNA is involved in both transcription and translation and also has catalytic and structural roles in the cell, knowing its structure is of fundamental importance since it will determine the function of the RNA molecule. We introduce a GA where a permutation is used to encode the secondary structure of RNA molecules. We discuss initial results on RNA sequences of lengths 76 and 785 nucleotides and present several improvements to the algorithm. We show that with a higher selection intensity through the Keep-Best Reproduction operator and 1-elitism the best results (i.e. the structures with the lowest free energy) are achieved.

1 Introduction

Many problems in AI and Optimization fall into the category of combinatorial optimization. In combinatorial optimization, a solution consists of n variables who can take m values each. This typically leaves $O(m^n)$ potential solutions. Typically not all solutions are equally good. For example, in the traveling salesman problem (TSP), not all tours have the same cost. Solutions to the TSP for n cities can be encoded as permutations of length n and we can think of the permutation as n variables holding one of n values each, with the additional constraint that no values can be assigned to no variable more than once. Several algorithms such as hill climbing (iterative improvement), threshold algorithms, simulated annealing and taboo search have been suggested to solve such optimization problems. Another optimization technique is Genetic Algorithms (GAs) where a population of candidate solutions is evolved, rather than searching in the neighborhood of only one candidate solution. The most important operators in genetic algorithms are Selection, Crossover, and Mutation. It is essential to keep a balance between exploration of the search space and its exploitation. In GAs, this is typically done by controlling the selection intensity and adding measures to maintain genetic diversity in the population. Fig. 1 depicts a high level genetic algorithm. Note that the stopping criteria can be defined by the user to be, for example, total CPU time, number of new solutions created, or other measures such as genetic diversity or a combination of measures.

RNA is essential for the functioning of a cell and is the product of gene transcription. In a process known as translation, the RNA is involved in the building of proteins.

```

Initialize a population of chromosomes;
Evaluate the chromosomes in the population;
while (stopping criteria not reached) do
  for i=1 to sizeof(population)/2 do
    select 2 parent chromosomes;
    apply crossover operator to them;
    apply mutation operator to them;
    evaluate the new chromosomes;
    insert them into  $g_{next}$ ;
     $i = i + 1$ ;
  od
  update stopping criteria;
od

```

Figure 1: A high level Genetic Algorithm

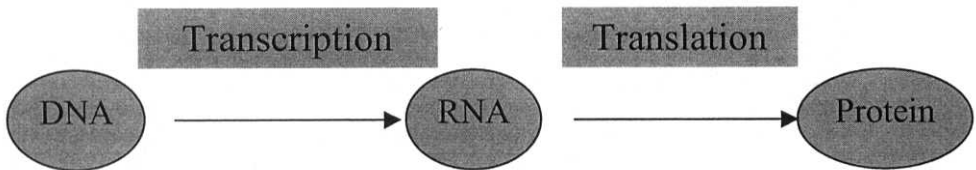


Figure 2: The role of RNA in Transcription and Translation

While DNA consists of a sequence of molecules made from the 4 nucleotides Adenine (A), Cytosine(C), Guanine (G), and Thymine (T), RNA consists of A, C, G, and Uracil (U), which replaces Thymine. In the translation process, a ribosome uses the additional tRNA available in the cell to produce a protein where 3 RNA nucleotides form a codon which encodes for one of the 20+ amino acids.

Besides being involved in these fundamental processes, RNA also has catalytic and structural roles in the cell. The structure of the RNA molecule is important as it determines its function. We can adequately describe the secondary structure of an RNA molecule as a set of stem loops consisting of a set of consecutive base pairs connected through hydrogen bonds.

VanBatenburg et. al. [10] have demonstrated that a GA can be used to predict the secondary structure of RNA and also to approximate the folding pathway that the RNA molecule may take during the folding process by adding and deleting stem loops. Their implementation used a binary representation for the solutions and they made several additions to the GA such as elitism that improved the results. Their initial algorithm had very poor performance and did not converge at all, due to the inherent incompatibilities of stem loops in the binary representation of the solutions. Subsequently, the authors suggest ways of “repairing” the solutions to yield feasible ones. A massively parallel GA for the RNA folding problem is discussed in [9] and an improved mutation operator is discussed in a subsequent paper [8]. In the latter paper the authors demonstrate that the improved GA is predicting more correct (true-positive) stem loops and more correct base pairs than what could be predicted with a dynamic programming algorithm (DPA). While DPAs can accurately compute the minimum energy within a given thermodynamic model, the natural fold of RNA is often in a sub-optimal energy state. It can be argued that the natural folding process of RNA and the simulated folding of RNA using a GA (which includes

intermediate folds) have many similarities, especially with regards to stem loop additions and deletions [10].

We propose to use a permutation based genetic algorithm to predict which helices (see exact definition in the next section) will form in the final secondary structure of the molecule. The objectives of this paper are as follows:

1. To introduce an alternative representation for RNA secondary structure that allows to use a permutation based GA for the RNA folding problem
2. To present an initial evaluation of the GA, including convergence behavior, efficiency and effectiveness
3. To study the effects of different selection strategies on the GA

In Section 2, we give an introduction to RNA structural elements. We discuss both the primary structure and the secondary structure of RNA and describe how the problem can be decomposed into a combinatorial problem of finding the sub-set of helices from a set of potential helices leading to a minimum energy in the molecule. We describe the permutation based GA that we use to solve the RNA folding problem in Section 3. Section 4 contains our empirical results. A discussion and conclusions are outlined in Section 5 and Section 6 respectively.

2 RNA Secondary Structure

The formation of RNA secondary structure is the result of a natural folding process in which chemical bonds (hydrogen bonds) form between canonical base pairs. It is important to note that the hydrogen bonds are not the cause of the folding, but rather the result. These hydrogen bonds can only form between certain pairs of nucleotides, i.e. AU, CG, GU, and their mirrors, UA, GC, UG, hence the name canonical base pair. Searching a nucleotide sequence for all possible base pairs is a fast and simple procedure. The difficulty comes in predicting which ones, out of all the possibilities, will form bonds.

Depending on which base pairs form bonds, various structural elements become evident in the resulting secondary structure. Hairpin loops contain exactly one base pair, internal loops contain exactly two base pairs, bulges contain exactly 2 base pairs with 1 base from each of its pairs adjacent in the backbone of the molecule. There are also multi-branched loops which contain more than two base pairs, and external bases which are not contained in any loop.

Stacked pairs, which form helices, provide stability in the secondary structure. A set of stacked pairs is formed by two or more base pairs $(i, j), \dots, (i+n, j-n)$ $1 \leq n \leq m$ such that the ends of the pairs are adjacent, forming a ladder type structure.

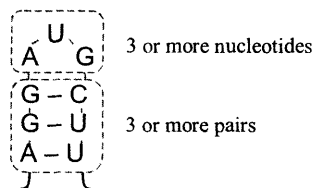


Figure 3: Rules for stem loop formation

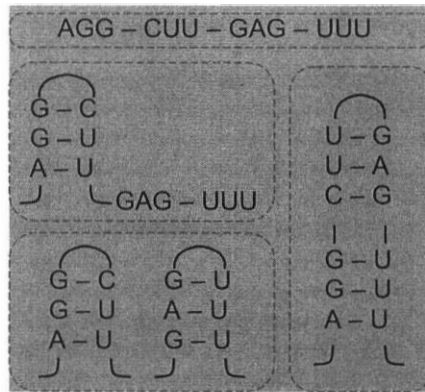


Figure 4: Sample sequence fragment and examples of possible helices that could form

Finding all possible helices made from these stacked pairs assists in determining the secondary structure.

There are certain requirements that must be met in order for a helix to form, as illustrated in Fig. 3. The helix must consist of at least three consecutive canonical base pairs forming a stack, and the loop connecting the two strands must be at least 3 nucleotides in length. These are chemical requirements for the stable formation of stem loops. Considering these constraints for the formation of helices, it is now possible to compute the set H of all helices that could potentially form. The solution to the secondary structure prediction problem is a subset S of H , which contains the helices that make up the final structure. Of course, care must be taken to only include helices in S that are mutually exclusive, i.e., do not share a nucleotide (again, chemically, such structures are infeasible). Since there are $2^{|H|}$ subsets of $|H|$, the problem is highly combinatorial. As Fig. 4 demonstrates, there are many possible helices from even a short sequence.

The technique employed to direct the GA through this large search space is energy minimization. As an RNA molecule will fold into a structure with near minimal energy, the GA attempts to find the combination of helices that result in the lowest energy possible. The energy function used in this implementation takes into account the stacking energies of the helices. A change in free energy (ΔG) is associated with each canonical base pair:

$$\begin{aligned}\Delta G(CG) &= -3 \text{ kcal/mol} \\ \Delta G(AU) &= -2 \text{ kcal/mol} \\ \Delta G(GU) &= -1 \text{ kcal/mol} \\ &(\text{all at } 37^\circ\text{C}).\end{aligned}$$

In order to calculate the energy of a given structure, the sum of energies of all pairs formed in the structure, is used:

$$E(S) = \sum_{i,j \in S} e(r_i, r_j).$$

Here, $e(r_i, r_j)$ is the free energy between the i^{th} and j^{th} nucleotide (residue) forming a base pair.

3 A Permutation Based GA to Predict the Structure

Originally, GAs were designed to work on bitstrings. The bitstring encoded a domain value of the real valued function that was subject to optimization. Theoretical justification for the convergence of GAs is provided by Holland's schema theorem [4].

By the mid nineteen-eighties, researchers had started to apply GAs to combinatorial optimization problems such as the TSP and the Job Shop Scheduling Problem (JSSP). For the TSP, a solution (a tour visiting each city exactly once and returning to the start city) was encoded as a permutation of the individual city numbers. For example, a permutation (3 4 2 1 5 6) means start at city 3, then go to city 4, then to city 2, and so on. Such a representation, however, is not suitable for standard "cut and paste" binary crossover operators since such operators would create offspring that are not permutations anymore due to duplications and omissions of cities. The literature contains operators that are suitable for permutations, i.e., will always create a permutation again. Also, these operators try to preserve and promote either absolute position of cities, partial order between cities, or adjacency between cities. Some of the operators discussed in the literature for direct representation include *partially mapped crossover* (PMX) [2], *order crossover* (OX) [1], and *cycle crossover* (CX) [6]. Similarly, for other ordering problems such as the JSSP a solution can be encoded as a permutation of orders for the varying machines on the shop floor.

As outlined in Section 2, the problem of predicting the secondary structure of RNA can be decomposed to become a problem of picking the sub-set S of helices from the set of all helices H , such that $E(S)$ is minimized and that no helices in S share a base pair. In order to use a GA to solve this problem we need to decide on a representation and operators. We propose the following:

If the set of all helices H contains n helices then use a permutation of length n to represent a candidate solution. The order in which a helix appears in the permutation is the order in which it is picked by the decoder to be inserted into the final structure. Helices that are incompatible with any previously selected helices are rejected. To illustrate this let us consider the following example:

- H is the set of all possible helices, $|H| = 6$
- $P = (3\ 4\ 2\ 5\ 6\ 1)$ is a candidate solution that is decoded as follows:
- **For** $i = 1$ **to** 6 **do**
 - if** $H[P[i]]$ has no conflict with any helices in S
 - then** add $H[P[i]]$ to S
 - fi**
- od**

This decoding algorithm will ensure that only valid secondary structures are produced by the GA. We can now run a GA using permutation based crossovers such as CX, PMX, and OX on the permutations which encode for secondary RNA structures.

Another operator that is critical for the performance of a GA is the selection operator. In our study we use both standard roulette wheel selection (STDS) and a strategy called Keep-Best Reproduction (KBR) introduced in [5] where parents are chosen by Roulette Wheel selection; then crossover and mutation are performed and the set of parents and offspring undergo an additional selection step where the fittest parent and the fittest offspring survive and are inserted into the next generation. The KBR operator was shown to increase both the efficiency and the effectiveness of the GA in the TSP domain and was also demonstrated to be more robust with regard to algorithm parameter settings.

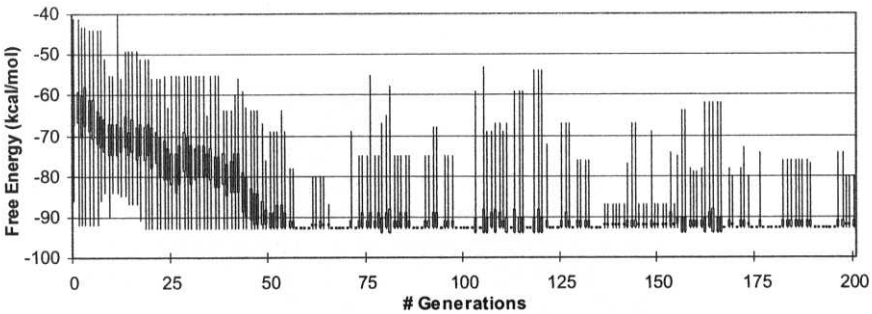


Figure 5: 90 Possible Helices, $P_m=5\%$, $P_c=70\%$, $pop_size = 50$, CX, STDS, no elitism

4 Results

4.1 Initial Evaluation

The initial results upon completion of a prototype appeared to be somewhat disappointing. During preliminary testing, a short 76 nucleotide tRNA sequence (*Saccharomyces cerevisiae* tRNA-Phe gene, GI:176479), resulting in 90 possible helices, was used. Fig. 5 demonstrates the relatively poor behavior initially encountered. The outer envelope of the plot shows the extremities of each generation (maximum and minimum energies present), while the darker, inner, envelope shows the mean free energy of the population with standard deviation. In this study we have used the CX operator with STDS using the following parameters: mutation probability (P_m) of 5%, crossover probability (P_c) of 70%, and a population size (pop_size) of 50 for 200 generations ($\#generations$). The small population size and few generations made this an ideal configuration for initial testing as a complete set of data could be obtained in a matter of seconds.

As can be seen, the structure with minimal energy found at the end of the run was only marginally better (-93 kcal/mol) than the structure with minimal energy of the initial, random population (-86 kcal/mol). Additional tests yielded a best solution of -97 kcal/mol. While the results did not significantly improve from the randomly generated initial population, there was evidence of encouraging behaviors. Most significant was the convergence behavior shown as the algorithm progressed through the generations. By examining the plot of the mean free energy of the population, it can be seen that there is a definite trending of the population towards structures with lower free energy. However, the population converges to a sub-optimal solution and loses its diversity very early (60 generations). Past this point, most small improvements come from the random mutation, which typically create weaker, rather than stronger, children as evidenced by the spikes from approximately generations 60 to 200. This effect is particularly pronounced with the STDS operator, as it does not examine the quality of the children before placing them in the future generation. The secondary structure with the lowest free energy found has only seven helices. This is a relatively small number and may have contributed to the relatively poor performance of the initial tests. In the next three subsections we will study our algorithm on a larger sequence of 785 nucleotides and the use of larger populations. We will introduce several improvements to the algorithm, which include the use of 1-elitism and the use of an improved selection strategy, namely Keep-Best Reproduction [5]. We will investigate several combinations of these strategies in order to determine which one is best.

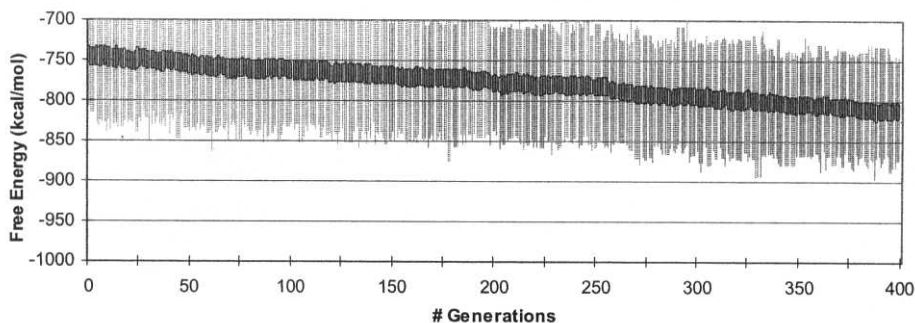


Figure 6: 10480 Possible Helices, $P_m=5\%$, $P_c=70\%$, $\text{pop_size} = 700$, CX, STDS, no elitism

4.2 Longer Sequence, Larger Population

In light of these findings, we decided to use a longer sequence as well as increase the size of the population and number of generations. We anticipated that this would increase the diversity of the population in order to avoid premature convergence as well as provide for more room for the algorithm to improve upon the initial, randomly selected, population. The parameters used for this run were largely the same as those used for Fig. 5 with the exceptions being a longer RNA sequence consisting of 785 nucleotides (homo sapiens mRNA containing U19H snoRNA, GI:2853215), a population size of 700, and a run length of 400 generations. As can be seen in Fig. 6, the envelopes of the maximum, minimum, and mean of the generations are significantly different from those in Fig. 5. Now the algorithm is able to improve from -809 kcal/mol to -873 kcal/mol within 400 generations, an improvement of $\sim 7.9\%$ which is almost identical to the previous improvement of $\sim 8.1\%$. As the mean clearly shows, there is a steady trend of the population towards structures with lower free energy. As well, the population remains quite diverse with very little loss of diversity. While there is progress towards a low free energy solution, the convergence is relatively slow, requiring many generations. Noteworthy is also that the best structure at generation 400 has a higher energy than some of the best structures found in previous generations. For example, a structure with $G = -896$ kcal/mol was found at generation 388, but was lost subsequently.

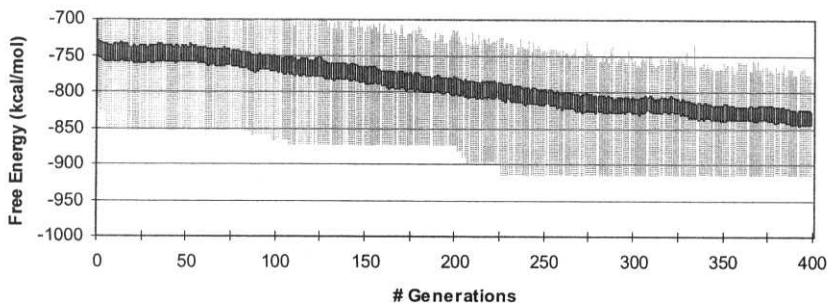


Figure 7: 10480 Possible Helices, $P_m=5\%$, $P_c=70\%$, $\text{pop_size} = 700$, CX, STDS, 1-elitism

4.3 Elitism

In order to preserve the best result from each generation, elitism was used to replace the structure with the highest free energy in generation i with the structure with lowest free energy from generation $i-1$. Results are displayed in Fig. 7. Due to the 1-elitism, the lower extremity of the graph is not as ragged as in Fig. 6 and a better solution with a free energy of -916 kcal/mol is found. Overall, this is an improvement of 13.2% over the best structure found in the initial random population.

4.4 Selection Strategies

The next step in testing was to use Keep Best Reproduction (KBR) to increase selection pressure. This increased selection pressure requires that the mutation probability be dramatically increased in order to help maintain diversity in the population. As a result, the only parameters different from the previous example are the mutation probability and the selection operator. P_m is now 80% rather than 5% and KBR is used instead of STDS. As is evident in Fig. 8, convergence is much faster with the KBR selection operator, arriving at better solutions sooner than with STDS. As a result of this rapid convergence, the diversity of the population is swiftly decreased and most benefits past 100 generations are the result of the extremely high mutation rate. However, this does not pose a problem for the algorithm. For example, after 50 generations KBR has a peak result of -899 kcal/mol whereas STDS has a peak result of -853 kcal/mol. After 100 generations, KBR has found a solution of -923 kcal/mol, while STDS has found a result of -866 kcal/mol. At the end of 400 generations, the KBR technique resulted in a solution of -980 kcal/mol, a $\sim 21\%$ decrease in free energy compared to the STDS decrease of $\sim 13\%$ at -916 kcal/mol.

5 Discussion

Further investigation is needed to assess how well a permutation based representation for the GA is suited for the prediction of RNA secondary structure. One issue is the over-representation of the search space. Many permutations that are different, encode the same solutions (after the decoder has rejected the incompatible helices). Also, currently, the helices kept are the ones that appear early in the permutation. Helices that appear later in the permutation and are incompatible with earlier helices are rejected. Using a heuristic that considers free energy contributions of helices that are incompatible could improve the

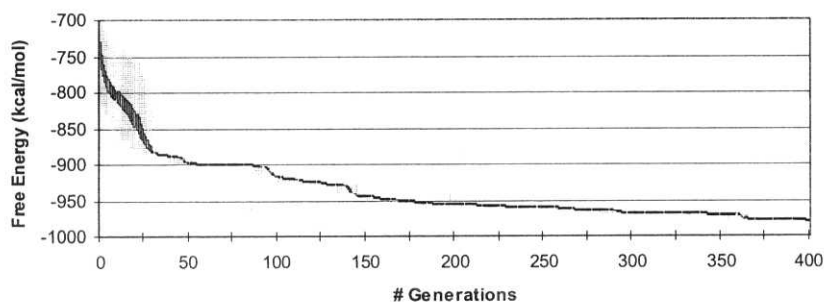


Figure 8: 10480 Possible Helices, $P_m=80\%$, $P_c=70\%$, $pop_size = 700$, CX, KBR, 1-elitism

performance of the GA further. For example, one could add a helix that appears later in the permutation and is incompatible with the previous selection if this addition leads to a decrease in free energy. Of course, in this case, we would have to delete any earlier helices that are incompatible with the new addition.

In addition, it would be prudent to study a binary version of the algorithm that uses a bit-string rather than a permutation to code for the solutions and do a one-on-one comparison. The challenge with a binary representation is that crossover frequently produces invalid solutions, but this can be solved by repairing the solutions. It will be interesting to see how the results of the binary algorithm would compare to the permutation based one.

Also, the study of other crossover operators and the study of larger sequences will reveal further insight into how well a permutation based encoding is suited for the RNA structure prediction domain. Ultimately, the predicted structures of low energy should be compared to the structures predicted by dynamic programming approaches and to known structures.

6 Conclusions

We have proposed a permutation based genetic algorithm to predict the secondary structure of RNA molecules using a simplistic energy function that accounts for the energy contribution of individual hydrogen bonds. The results on a sequence of 785 nucleotides demonstrate a good convergence behavior of the algorithm and the GA is able to improve the initial random solutions substantially. 1-elitism has proven useful for both STDS and KBR, where, clearly, STDS benefits more from elitism than KBR. The best results overall were achieved with the KBR selection operator (both with and without elitism) and even with elitism, STDS performs worse than KBR without elitism. With KBR, both the efficiency and the effectiveness of the algorithm increase substantially.

It is interesting to note that while the worst solution and the average solution in the population show an exponential convergence behavior, the curve for the best solution in the population is rather that of a stepwise linear function for STDS. Another observation is that the best solution in the initial population already has a relatively low free energy. This was especially observed on shorter sequences.

7 References

- [1] Davis L. (1985), "Job Shop Scheduling with Genetic Algorithms", *Proceedings of the 1st International Conference on Genetic Algorithms ICGA-85*, pp. 136-140
- [2] Goldberg D.E., Lingle R. Jr. (1985), "Alleles, Loci, and the Traveling Salesman Problem", *Proceedings of the 1st International Conference on Genetic Algorithms ICGA-85*, pp. 154-159
- [3] Gulyaev AP, van Batenburg FHD, and Pleij CWA (1998), "RNA folding dynamics: Computer simulations by a genetic algorithm", *Molecular Modeling of Nucleic Acids, ACS Symposium Series*, 682, pp. 229-245
- [4] Holland J.H. (1975), *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI, Michigan University Press
- [5] Wiese, Kay C. and Goodwin, Scott D. (2001), "Keep-Best Reproduction: A Local Family Competition Selection Strategy and the Environment it Flourishes in", *Constraints*, Vol. 6, Kluwer Academic Publishers, pp. 399-422
- [6] Oliver I.M., Smith D.J., and Holland J.R.C. (1987), "A Study on Permutation Crossover Operators on the Traveling Salesman Problem", *Proceedings of the 2nd International Conference on Genetic Algorithms ICGA-8*, pp. 224-230
- [7] Shapiro B.A., Wu J.C., Bengali D., Potts M.J. (2001), "The massively parallel genetic algorithm for RNA folding: MIMD implementation and population variation", *Bioinformatics*, 17(2), pp. 137-148

- [8] Shapiro, B. A. and Wu, J. C. (1996), "An annealing mutation operator in the genetic algorithms for RNA folding". *Comput. Appl. Biosci.* 12(3): 171-180
- [9] Shapiro, B. A. and Navetta J. (1994), "A massively parallel genetic algorithm for RNA secondary structure prediction", *Journal of Supercomputing*, 8(3), pp. 195-207
- [10] VanBatenburg F.H.D., Gulyaev A.P., and Pleij C.W.A. (1995), "An APL-programmed Genetic Algorithm for the Prediction of RNA Secondary Structure", *Journal of theoretical Biology*, 174(3), pp. 269-280

Design of Strong Causal Fitness Functions

S. Hirche, I. Santibanez-Koref, I. Boblan

TU-Berlin; FG Bionik u. Evolutionstechnik; Ackerstr. 71-76; 13355 Berlin; GERMANY

Abstract. A new kind of fitness functions for controller optimization is presented. This new fitness functions are postulated to be strong causal. Thus a better behaviour during the optimization process can be achieved.

1 Introduction

The design of dynamical systems for industrial applications can be divided into two parts: first the identification of a plant or system and second the design of an appropriate controller.

This paper presents an approach to the controller design problem. The paper focuses on the problem of parameter identification. We use an Evolution Strategy (ES) to identify the parameters [12, 14].

We implemented this approach in the SCADS system (Strong Causality driven generation of Dynamical Systems, [13]). The main purpose of SCADS is to give a framework for automatic plant and controller design. One property of SCADS is that one basic concept behind the implementation is the use of "Strong Causality" [12] through all stages of the search process. "Strong Causality" means that small variations of the individual, i.e. dynamical system, should produce small variations of the quality.

Genetic Programming (GP) [7] is used for the structure identification of the dynamical system. The GP is adapted by implementing strong causality in the representation and in the genetic operators [11]. The ES is used to adapt the parameters of the dynamical systems. Strong causality in this algorithm means that the quality assignment to each system should be strong causal.

Another aspect of SCADS is the practical usability. That's why we build it up using Mathworks Inc. MATLAB/SIMULINK. MATLAB offers a wide variety of tools for the design and identification of dynamical systems. Specially a number of toolboxes offers well proven applications and tailored methods for the simulation of dynamical systems.

In order to use SCADS, we have to formulate an appropriate description of dynamical systems. That means that the description should be evolvable and be able to describe a wide variety of dynamical systems. We choose a presentation as a directed graph [11]. This special representation as a graph guarantees that it is possible to create closed loops in the dynamical systems, thus it is possible to generate the most common structures for dynamical systems. The nodes represent the standard building blocks for dynamical systems (e.g. P-, PD-, PID-blocks etc. [2]). The vertices represent the signal flow between these blocks. One individual is a dynamical system, represented as a graph. That means, that we are evolving graphs.

As an important feature it is necessary to assign to each individual a quality or fitness. In our case quality means: How well does the dynamical system fulfill the required task. In case

of system identification, how well does it reproduce a given behavior. In the case of controller design: how well does the system control a given plant. In order to assign a suited quality the user has to provide a calculation procedure, which involves the simulation of the dynamical system by a integration algorithm. As we use standard representation for functional blocks, we have to take in to account that the blocks are parametrized. Thus not only the structure of the dynamical system is important, also the parameters of the blocks are significant for the resultant system behavior. We have to choose the appropriate parameter settings in order to decide what is a good system. Thus it is necessary to optimize the parameters in order that the quality criterion is fulfilled as best as possible. That means that besides a simple simulation of the dynamical system is it indispensable to optimize the parameters of this system in order to determine the quality of the system.

GP will produce dynamical systems which will be hard to optimized. The reason can be that the produced systems have unfavourable numeric properties (e.g. stiffness) or the start parameter settings will be distant from useful values. For these reasons we chose an ES as robust optimizer [12, 14]. Other optimization algorithms, in particular deterministic ones, are known to be faster under certain conditions, which are not guaranteed to fulfilled by GP generated dynamical systems.

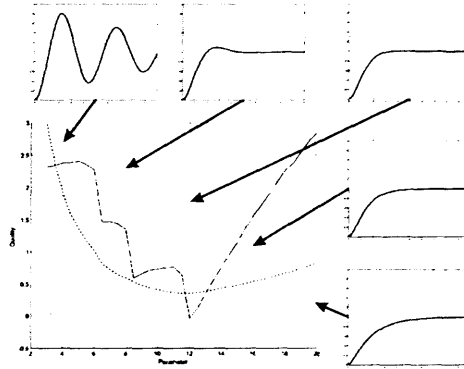


Figure 1: Comparison between a Strong Causal (dotted line) and a non Strong Causal quality function for simple control system.

Providing the quality information it is easy to select the best individuals. The first choice would be to use truncation selection or tournament selection. But taking in to account that for each quality assignation a simulations are needed, we decide to choose another selection structure. For selection we take a certain number of individuals from the population, select the best from this group and replace the worst individuals by new generated individuals (called partial tournament selection). Thus we can guarantee that we have not a generational EA. That guarantees a better usage of available parallel resources. Using this approach for selection we can expect an efficient implementation of the EA. That is why we implement the quality assignment on independent, asynchronous computational slaves, which guarantee a linear speed up of the execution time.

The next problem is how to define a strong causal quality function. This quality is decisive for the efficiency of the whole algorithm. Figure 1 presents two functions for quality assignment for controlled system, in which the controller depends on one parameter. The

continuous line represent a quality function that tends to have big value changes with small parameter changes. The function is multimodal and hard to optimize for a wide variety of optimization algorithms. The second function depicted in figure 1 is a so called strong causal function (dotted line). We can see that the function is smoother, thus small changes in the parameter gives also small changes in the quality. The second property of this function is that it is unimodal, thus it can be optimized very easy.

In this paper we describe one variant of the quality assignment for dynamical systems in SCADS.

The structure of the paper is as follows: In the next section we give an overview over the classical quality criteria for controllers. In section 3 we present the new controller criteria. Some results obtained with this criteria are described in section 4. Conclusions are drawn in section 5.

2 Quality Criteria for Controller Optimization

Most quality criterions known from the practice of control system design, e.g. the characteristic value criterion [15] and the integral criterion value the transient response characteristic to a test input signal in the time domain as shown in Figure 2. Here $r(t)$ stands for the reference input, $e(t)$ is the error signal, $u(t)$ symbolises the correcting variable and $y(t)$ is the output signal of the system.

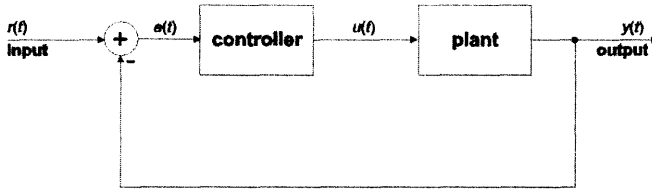


Figure 2: Basic control system with unity feedback

Fortunately the performance indices defined in time-domain [4] can easily be measured. They result directly from the application and can be considered as standard. Usually a step input is used for reference input [2].

- The swiftness of the response is defined by the rise time t_r and the peak time t_p . For an over damped system the peak time is not defined and the 10%-90% rise time t_r is normally used (see [2]).
- The similarity of the response $y(t)$ to the step input $r(t)$ is represented by the peak value or overshoot M_p and the settling time t_s . The settling time t_s is defined as the time required for the system to settle within a certain percentage of the input amplitude (see [2]).

In the characteristic value criterion the interesting performance indices are summarized. This method provides a transparent correlation between the transient response characteristic and the fitness function. But the fitness landscape turns out to have ridges. That reduces the

convergence speed or even worse, the optimisation algorithm can get stuck there. Therefore this criterion is not suitable for numerical optimisation [10].

The integral criterion provides smooth fitness landscapes by valuating the quality using the weighted squared error that is evaluated between a reference function and the transient response to a test input signal.

$$Q_I = \int_0^T f(e(T), u(t), y(t), t) dt$$

So in this criterion the location of the optimum can be influenced (moved) by means of the reference function choice, the time weighting and the weighting of the correcting variable. As reference signal a unit step input is often used. A widely applied formulation of the general integral criterion is the integral of time multiplied by the squared error criterion (ITSE):

$$Q_I = \int_0^T t^n (e^2(T) + \alpha u(t)) dt$$

The time weighting causes a higher penalty the later a deviation from the reference signal occurs. The effect is that the optimisation leads to more stable systems, at least within the time horizon. The squared correcting variable $u(t)$ is multiplied with a scalar weight α . In theoretical developments the problem of the correcting variable is often ignored and without importance. In real-world problems though the limitation of the correcting variable must be taken into consideration.

This approach has two main disadvantages. Firstly, the solutions are often not definite [10] and meet the performance requirements often inaccurately, since the aspired aim is not to optimise towards a desired characteristic value but to minimise the squared error. Additional penalty terms for certain characteristic values do not provide appropriate results either [1]. A simple way to transfer the performance indices into the criterion is by comparing the output signal with a reference signal that meets the desired performance requirements, e.g. with PT2-behaviour. Consequently the optimal control system in terms of such a performance measure never behaves better than the compared PT2-system. This is an unacceptable restriction.

Secondly, as the correcting variable is minimised in relation to the signal error, thereby the solutions are often conservative. In most cases this formulation does not reflect the situation of real-world problems, where the correcting variable is bounded and therefore set to the maximum in order to speed up the transient response.

Subsequently a quality criterion is designed that combines the advantages of both performance measures: the implementation of the time-domain constraints and a satisfying smooth fitness landscape without ridges.

3 Alternative quality criterion

The basic idea is, to divide the general form of the integral criterion into signal value orientated time sections [1]. We call it piecewise integral of time multiplied by the squared error criterion or short PITSE. By using the characteristic values an indirect avoidance of parameter ambiguity can be reached. Each section represents one characteristic value of the

step response with the magnitude r . The more values need to be considered the more sections have to be created. In most cases four characteristic values thus sections prove to be sufficient. Some of them are signal value related (M_p, M_U, e_{ss}) others time related (t_r, t_s). In order to exert a comparable selection pressure by each of the time sections the dimensions of them must match. All sections are based on the integration of a squared error. The differences can be found in the measurement of the error and the integration boundaries. Another point regarding the optimisation needs to be taken into account. The start vector of the control parameter is often set to 1 and accordingly leads to a system response that never lies in the near surrounding of the desired one. In order to achieve a flow transition from far to near optimisation the objective function must meet the demand for Strong Causality as mentioned above. Therefore high values are assigned to each section if the solution is still far from the optimum, so in terms of control systems if it is unstable. The objective function is not changed in the course of approaching the optimal solution. The fitness function consists of the following sections:

Undershoot section: Preventing an undershoot is often impossible but minimising is required. The difference $e_{mu}(t)$ between the signal and the boundary value for undershoot $M_{u,req}$ is squared and integrated over the entire time horizon $[0, T]$:

$$q_1 = \int_0^T e_{mu}^2(t) dt \quad e_{mu}(t) = \begin{cases} 0 & \text{for } y(t) > M_{u,req} \\ y(t) - M_{u,req} & \text{otherwise} \end{cases}$$

As soon as the system gets unstable, the part q_1 can reach high value. Thereby this section supports the movement towards stable systems.

Rise time section: In control system design the requirements for the rise time and the overshoot often turn out to be the most important aspects after the demand for robust stability. Again a squared error $e_{tr}(t)$ is integrated within the boundaries ($t_{r,req}, t_1$) where $t_{r,req}$ is the required rise time and the value of t_1 depends on the step response characteristic:

$$q_2 = \int_{t_{r,req}}^{t_1} e_{tr}^2(t) dt \quad e_{tr}(t) = \begin{cases} 0 & \text{for } t |_{y=r} \leq t_{u,req} \\ r(t) - y(t) & \text{otherwise} \end{cases}$$

$$t_1 = \begin{cases} t |_{y=r} & \text{for } y(t) \geq r \\ T & \text{otherwise} \end{cases}$$

The error $e_{tr}(t)$ is defined by the difference between the amplitude of the step input $r(t)$ and the signal $y(t)$. If the signal does not cross the zero line within the time horizon, a rise time is not defined at all. By double counting the undershoot area a solution far from the optimum is highly penalized in order to increase the selection pressure and therefore speed up the searching. Additionally a smooth transition from far to near optimisation is reached. In Figure 3 several scenarios for the error evaluation of the rise time section are shown.

Overshoot section: In correspondence to the undershoot section the overshoot section is designed. The squared error $e_{Mp}(t)$ is defined as the difference between the required maximum of overshoot $M_{p,req}$ and the signal $y(t)$ within the time horizon $[0, T]$ is integrated:

$$q_3 = \int_0^T e_{Mp}^2(t) dt \quad e_{Mp}(t) = \begin{cases} 0 & \text{for } y(t) < M_{p,req} \\ M_{p,req} - y(t) & \text{otherwise} \end{cases}$$

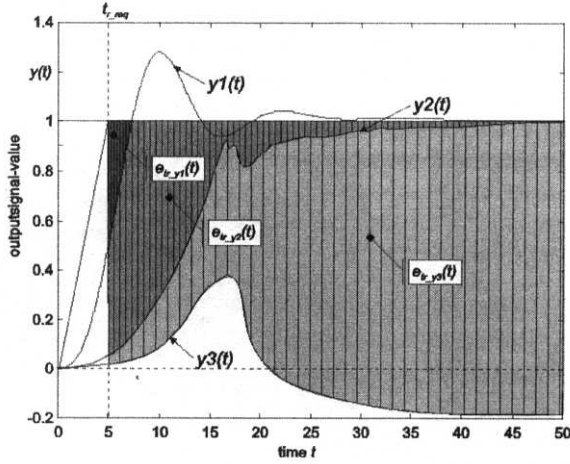


Figure 3: Scenarios for the evaluation of the rise time error $e_{tr}(t)$

Steady state error section: The steady state error and therefore the static system behaviour is valued by this part of the fitness function. If the signal leaves the ϵ -tube after the required settling time t_s , then the difference $e_{ts}(t)$ between the ϵ -tube and the step response is squared, time weighted and integrated:

$$q_4 = \int_0^T e_{ts}^2(t) dt \quad e_{ts}(t) = \begin{cases} 0 & \text{for } -\epsilon \leq \frac{y(t)}{r(t)} - 1 \leq \epsilon \\ y(t) - t(t) \cdot (1 + \epsilon) & \text{for } y(t) > r(t) \cdot (1 + \epsilon) \\ y(t) - t(t) \cdot (1 - \epsilon) & \text{otherwise} \end{cases}$$

Systems with an unstable step response are highly penalized. The time weighting indirectly leads to stable systems. By means of the time weighting this part of the fitness function becomes the main power exerting selection pressure far from the optimum. Additionally the triple evaluation of under-, overshoot and steady state error in case of unstable solutions accelerates the far optimisation.

Correcting variable term: In real-world problems the value of the correcting variable is physically bounded within a lower value u_{low} and an upper value u_{up} . As long as the correcting variable remains within these boundaries this part maintains the value 0. Whenever the correcting variable crosses a boundary, an error function $e_u(t)$ is defined. Consistent with the evaluation of the other sections the error is squared and integrated:

$$q_5 = \int_0^T e_u^2(t) dt \quad e_u(t) = \begin{cases} 0 & \text{for } u_{low} \leq u(t) \leq u_{up} \\ u(t) - u_{up} & \text{for } u(t) > u_{up} \\ u_{low} - u(t) & \text{otherwise} \end{cases}$$

Unstable systems tend to have high values of the correcting variables; therefore unstable systems get penalized by this term, too.

There exist several possibilities to combine these sections to a scalar fitness value [3]. In the following the quality is presented by the sum of the weighted section values q_i :

$$Q_{PITSE} = \sum_{i=1}^5 \alpha_i \cdot q_i$$

The advantage of this method compared to others is its efficiency. As weakness appears the difficulty to find the appropriate weights α_i in order to reach a desired solution. Therefore an iterative adjustment process of the weights in the course of optimisation turns out to be necessary. Some of these sections are in concurrence to each other, e.g. the rise time section versus the overshoot section. In order to gain a lower rise time an increase of overshoot must be accepted. A compromise must be obtained which leads to the problem of vector or multi-objective optimisation [3]. The weighting can influence the balance of the trade-off. Unlucky weighting can distort the relationship between the sections and thereby it can cause the getting stuck of the ES in a local optimum. In most cases satisfying results are achieved by setting all weights to 1 in the new fitness function. In case that all performance requirements are met, the fitness function obtains the value 0, which is a terminating condition for the ES. In order to continue the optimisation the required characteristic values must be set to a lower value. Even if a well balanced selection pressure concerning each penalty term thus characteristic value was aspired, several simulations have shown that the rise time term exerts the highest selection pressure followed by overshoot-, undershoot term, the correcting variable term and last the steady state error term. The optimum step response tends to meet the required rise time rather than the required overshoot or the required settling time with all weights set to 1. But this depends on the system and the length of simulation, too. In order to lay the emphasis of optimisation to these characteristic values, the weighting must be changed. The optimisation of a system with unknown behaviour is an iterative process. The new criterion guarantees a steady selection pressure over the whole variety of output signals. This is absolutely necessary, because if the ES finds a local gap, the convergence speed decreases or the algorithm even gets stuck in multimodality.

4 Simulation results

In the following simulations the behaviour of new quality criterion is studied. The results of the optimisation with ITSE and PITSE are compared. All simulations are accomplished in the same environment under MATLAB/SIMULINK with the stiff variable step solver 'ode23tb' [9]. For the optimisation the (3,10) ES-CMA [5] with the start parameter and the start step size of 1 is used. First of all the behaviour of the single sections in optimisation is studied separately on a test system shown in Figure 4 a). Therefore all weights are set to zero except the one for the studied part of the quality function, which is set to one. With the start parameter settings the unit step response has unstable behaviour with undershoot as shown in Figure 4 b) (solid line). The parameters of the PID-Controller are given in the K-normal form [4]. Different test cases are regarded:

Undershoot section: Now the unit step response is required to have no undershoot, $M_{U,req} = 0$. In the test system there is no reasonable possibility of preventing an undershoot. The only solution to obtain the desired performance is the theoretical one by setting all parameters to zero causing a system output of zero magnitude. Surely this is of no practical

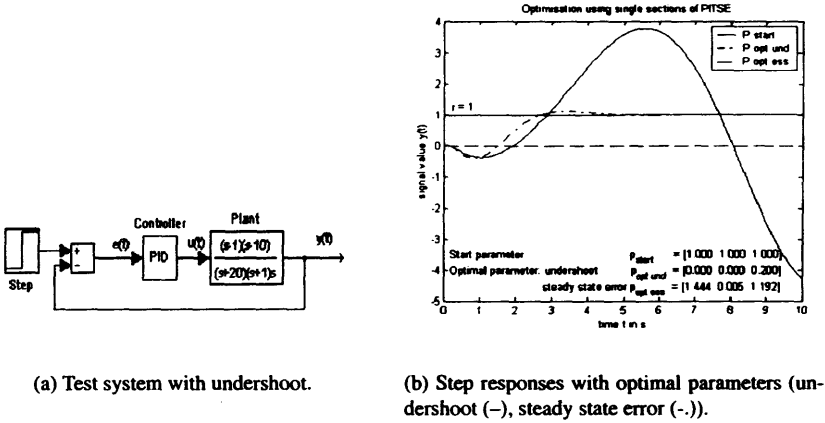


Figure 4: Test system with undershoot

use, but shows the effect of the undershoot section. After 60 generations the terminating condition for the quality function value of $Q = 10^{-10}$ is reached. The optimal parameter setting leads to an output that is close to zero shown in Figure 4 b) (dashed line). The signal value remains within the interval of $[-10^{-4}, 10^{-4}]$.

Steady state error section: Now the unit step response is required to remain within the 2%-band of the desired value $r = 1$ after the required settling time $t_{s,req} = 5s$. The terminating condition for the fitness value is reached after 60 generations. The optimal control system step response shown in Figure 4 b) (dash-dotted line) does not leave the 2%-band after the desired settling time.

Rise time section: The optimal parameter setting for the required rise time of $t_{r,req} = 2s$ is reached after 5 generations. The generated system is unstable, but the rise is below the requirements desired rise time.

Overshoot section: After 6 generations the optimisation is terminated with a fitness value $Q < 10^{-10}$. The step response with the optimal parameter setting has an overshoot smaller than the required $M_{p,req} = 1.2$.

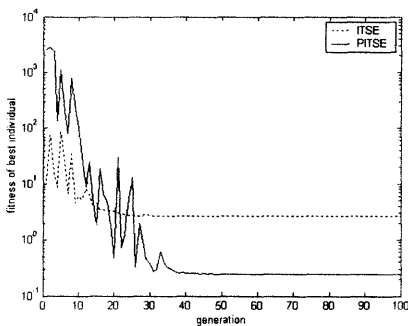
Now it is shown that each error section optimises towards the required characteristic value. In the following simulations the single error sections are combined to a fitness value as proposed in the previous section. The results of the optimisation with PITSE and ITSE are compared. The goal is to obtain an optimal step response with acceptable undershoot $M_{u,req} = -0.2$, rise time $t_{r,req} = 2s$, no overshoot $M_{p,req} = 1$ and settling within the 1%-band after the settling time $t_{s,req} = 5s$. The correcting variable value should remain within the boundaries of $[-5, +5]$. In Table 1 the results for the best of 10 runs are presented. In the third column the optimal parameters are listed, in the forth one the fitness value evaluated by the PITSE-criterion. The subsequent columns show the performance indices of the step response. In the last column the number of generations is listed when the fitness value enters the $\pm 5\%$ -band of the fitness value in the optimum. This parameter is used to indicate the convergence speed of the optimisation algorithm.

The optimisation by ITSE with the correcting variable weight set to $\alpha = 0.1$ creates a system with a rather conservative step response expressed by a high rise time, little overshoot and a low settling time. The emphasis lays on reducing the error of the latter part of the signal. The first runs with the PITSE-criterion with all weights set to one show very low rise time, caused by a higher correcting variable value. The performance concerning the undershoot is unacceptable though (2. row). In the following runs the weight for the undershoot section is set to 10 in order to gain lower undershoot (3. row). The undershoot is reduced to the level of optimisation with ITSE, but the signal does not settle within the simulation time. Therefore in the next optimisation (4. row), the steady state error section is weighted by $\alpha_4 = 100$. Now the performance of the system seems acceptable, even if the undershoot is still pretty high. The undershoot cannot be reduced without a trade-off concerning the rise time:

Table 1: ITSE and PITSE with variable weighting

	Weights	PID	Q	M_u	t_r	M_p	t_s	$ u_{max} $	Gen
ITSE	$\alpha_1 = 0.1$	[1.334 0.003 1.333]	0.161	-0.391	6.040	1.005	5.248	2.689	29
PITSE	$\alpha_1 = 1$	[1.474 -0.003 1.504]	0.046	-0.480	2.400	1.022	7.067	3.307	57
PITSE	$\alpha_1 = 10$	[1.394 -0.006 1.293]	0.109	-0.402	3.098	1.002	> 10	2.760	38
PITSE	$\alpha_1 = 10,$ $\alpha_4 = 100$	[1.388 0.001 1.286]	0.113	-0.400	3.028	1.009	5.140	2.743	39

The last column of Table 1 and the evolution of the fitness presented in Figure 5 a) indicate faster optimisation by means of the old integral criterion. The ITSE-criterion provides a smooth fitness landscape, whereas multi-objective fitness functions usually create multi-modal fitness landscapes that lead to lower convergence speed. From the step responses in Figure 5 b) it can be seen, that the faster response of the PITSE-optimal system results from the higher level of $|u(t)|$ in the rise time section. Here the conservative character of optimisation with the ITSE-criterion becomes obvious.



(a) Optimisation with ITSE and PITSE.

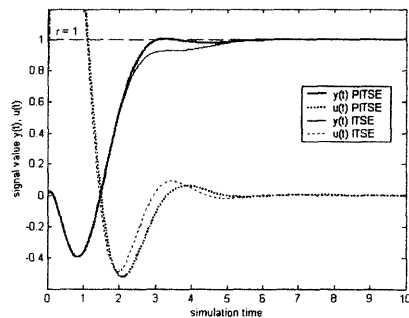
(b) Optimal Step responses of $y(t)$ and $u(t)$.

Figure 5: Optimization results of the different criteria.

5 Conclusions

The principal contribution of this paper is the introduction of an alternative quality function for parameter optimisation with Evolution Strategies, combining the advantages of the ITSE criterion and the characteristic value criterion. The integral criterion is divided into error sections that are adjusted on the characteristic values of the signal in the time domain. A smooth fitness landscape is achieved. Time-domain based, user-specified performance requirements are implemented and determine the location of the optimum in the parameter space. By means of weighting the priority of reaching certain performance specifications in the optimum can be defined. The new criterion is applicable to all control system design problems with time domain based performance requirements.

References

- [1] I. Boblan, I. Santibáñez Koref, A. Schütte: "A New Integral Criterion for Parameter Optimisation of Dynamic Systems with Evolution Strategy". VDI Berichte Nr. 1526 pp. 143 - 150. 2000
- [2] R.C. Dorf, R.H. Bishop: "Modern Control Systems". Prentice Hall. 2000
- [3] C.A. Coello Coello: "A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques". 1998
- [4] O. Föllinger; F. Dörrscheidt, et al.: "Regelungstechnik: Einf. in d. Methoden u. ihre Anwendung." Heidelberg, Hüthig-Verlag. 1994
- [5] N. Hansen and A. Ostermeier: "Convergence Properties of Evolution Strategies with the Derandomized Covariance Matrix Adaptation: The $(\mu/\mu_I, \lambda)$ -CMA-ES". 5th Europ. Congr. on Intelligent Techniques and Soft Computing: S. 650 - 654. 1997
- [6] S. Hirche: "Strong Causal Quality Functions for automatic Controller Design", Beiträge zur Bionik und Evolutionstechnik 02/2001, Technical University Berlin, ISSN 1618-8675, Berlin. 2001
- [7] J.R.Koza: "Genetic Programming: On the Programming of Computers by Means of Natural Selection". MIT Press. 1992
- [8] J.R. Koza et al.: "Automatic creation of human-competitive programs and controllers by means of genetic programming". Genetic Programming and Evolvable Machines. 1 (1 -2) pp. 121 - 164. 2000
- [9] MathWorks Inc.: "SIMULINK : dynamic system simulation software". Natick, Mass., Math-Works Inc. . 1994
- [10] M. Meyer: "Parameteroptimierung dynamischer Systeme mit der Evolutionsstrategie". Berlin, TU. Berlin. 1998
- [11] J. Niehaus, W. Banzhaf: "Adaption of operator probabilities in genetic programming". In Julian F. Miller, Marco Tomassini, Pier Luca Lanzi, Conor Ryan, Andrea G. B. Tettamanzi, and William B. Langdon, ed., Genetic Programming, Proceedings of EuroGP'2001, volume 2038 of LNCS, pp. 325-336. Springer-Verlag. 2000
- [12] I. Rechenberg: "Evolutionsstrategie 1994". Fromann Holzbog. Stuttgart. 1994
- [13] F. Lohnert, A. Schütte, J. Sprave, I. Rechenberg, I. Boblan, U. Raab; I. Santibanez Koref, W. Banzhaf; R. Keller; J. Niehaus, H. Rauhe: "Genetisches Programmieren für Modellierung und Regelung dynamischer Systeme - GEPROG -". Beiträge zur Bionik und Evolutionstechnik 01/2001, Technical University Berlin, ISSN 1618-8675, Berlin. 2001
- [14] H.-P. Schwefel: "Evolution and optimum seeking". New York, Wiley. 1995
- [15] H. Unbehauen: "Regelungstechnik I, Klassische Verfahren zur Analyse und Synthese linearer kontinuierlicher Regelsysteme". Vieweg Verlag. 1997

Noise and Elitism in Evolutionary Computation

Tuvik Beker and Lilach Hadany

Center for Neural Computation, the Hebrew University, Jerusalem, Israel
becket@alice.nc.huji.ac.il

School of Mathematical Sciences, Tel-Aviv University, Tel-Aviv, Israel

Current address:

Department of Biological Sciences, Stanford University, CA 94305, USA.
lilach@charles.stanford.edu

Abstract. Evolutionary Computation applications often involve a high degree of noise in the evaluation of solution performance. In the Genetic Algorithms literature, this has usually been considered a limitation, and attempts to reduce this noise seem to be the common practice. After introducing a proper measure for the performance of evolutionary techniques, we show that for complex fitness landscapes, noise reduction is not necessarily helpful, and sometimes is in fact harmful. Based on insights gained from biology, we suggest a simple form of Elitist selection, termed *Group Elitism*, which has a strong and robust positive effect on the performance of noisy algorithms. We discuss the notion of Elitism as an extreme form of *Fitness-Associated Variation*, and show that Group Elitism efficiently deals with the harmful effects of evaluation noise without compromising its benefits in evading local adaptive maxima.

1 Introduction

Uses of Evolutionary Computation often involve noisy fitness evaluation, where the fitness of individuals is assessed with a limited precision level. Evolutionary development of Autonomous Agents is a typical demonstration of this problem: the evaluation of agent fitness is subject to stochasticity in its environment, and is thus inherently very noisy. Noisy evaluation increases the survival probability of 'lucky' inferior solutions on the account of superior 'unlucky' ones, and has thus been traditionally considered harmful (see [5, 8, 16, 17, 2], but also [6] for a somewhat different view). The classical line of research compared multiple-sampling to population enlargement as means of overcoming the evaluation noise, studied convergence properties under noise in simplified models, and estimated population-sizing parameters.

In this paper we shall use a performance measure which is particularly suited to assessing the performance of evolutionary techniques, and takes into account the large variance between different simulation runs. We will show that in many realistic scenarios noise reduction is not only impractical, but may in fact have a negative effect on performance. Instead, we study a method based not on the reduction of noise, but rather on re-distribution of variation in the population. In its extreme form, this idea is very close to the well-known idea of elitist selection. We will show that this methodology, which carries a negligible computational cost, is of particular merit when dealing with noisy evaluation. Results will be demonstrated for a realistic engineering problem - the evolution of Neuro-controlled Autonomous Agents, as well as for a set of synthetic benchmark functions.

2 Evolutionary Performance and Computational Efficiency

Two common statistics used to measure the performance of evolutionary techniques are the average population fitness and the average over a certain top percentile of the population (both averaged over many simulation runs). These measures, though giving an indication about the performance of single runs, fail to measure the performance of one evolutionary technique (or parameter set) compared to another. This is primarily due to the high variability between runs using the same technique and parameters.

In a typical application of Evolutionary Computation, the main interest is not in the mean performance obtained but rather in the Champions. Many bad solutions can be tolerated as long as at least one very good solution is obtained. In the following we will use a more appropriate measure of evolutionary performance - the percentage P_θ of runs in which the best obtained solution passed a certain performance threshold θ . Taking an increasing series of thresholds allows for an intuitive comparison between different techniques, which answers the basic question of the experimentalist: "Which technique is most likely to obtain the best single solution for a given computational price?". When comparing different evolutionary techniques with a similar overall computational cost per simulation run, higher P_θ values are thus the natural indication of superior performance.

3 Methods

As a realistic test case we used a complex evolutionary simulator for Neuro-controlled Autonomous Agents. The full description of these simulations is beyond the scope of this article, and can be found in [1]. We will hereby describe some of their key features.

Agents that have to cope with a food gathering/poison avoidance task were evolved in a simulated grid arena. The agents were equipped with a fixed set of sensory and motor capabilities, intermediated by an evolved Artificial Neural Network. They were able to sense their immediate surroundings, make turns to the right or left, move forward, and eat resources which could have either a positive or a negative value. The weights of the fixed-size, fully-recurrent control network were encoded in a real-valued genome of several hundred bytes. The genetic operators used were uniform crossover with probability 0.35 and delta-type mutation with a small fixed delta value and a mutation probability of about 3% per site. Early experiments with the simulator proved that the results were quite robust to changes in the parameters for mutation and recombination. The values used were obtained after an initial sparse sampling of the parameter space. Each generation, 100 agents were evaluated, each in a separate environment, with randomly scattered food and poison. The fitness score was the number of food items eaten minus the number of poison items consumed within 150 action steps, normalized to a maximum value of 1. The selection method used was Roulette-Wheel type. To obtain accurate fitness measures for the population of the last generation, each agent in this generation was evaluated 1000 times.

In addition to the agent evolution task, we used a set of six synthetic benchmark functions, summarized in table 1. The suite consisted of the five De Jong benchmark functions [12] and the generalized function by Ackley [2]. To each of these functions we added a Gaussian noise term¹, with variance σ^2 . All the functions in the benchmark suite have a global minimum with value 0 at $\vec{x} = 0$, and the objective is minimizing the functions (see [7], chapter 4). The results presented here used the following parameters: population size 100, mutation rate 0.01 per genome, recombination rate 0.5. The results were robust to changes in all of these parameters.

¹Function f_4 contains a Gaussian noise term by its original definition.

Table 1: The benchmark suite

Fn.	Dimension	Bits per coordinate	Description
f_1	3	10	Paraboloid
f_2	2	12	Rosenbrock's Saddle
f_3	5	10	Step function
f_4	30	8	Noisy weighted x^4
f_5	2	17	Shekel's Foxholes
f_6	30	8	Ackley's Function

4 Noisy evaluation revisited

The most obvious effect of noisy fitness evaluation is that good solutions are likely to perish in favor of merely 'lucky' ones. The common intuition seems to predict a negative impact caused by evaluation noise, which is indeed the case for simple objective functions, such as the ones often used in analytic treatment of the subject [11, 16]. Solution re-sampling (i.e. averaging over multiple evaluations of each solution) and population enlargement were commonly used to counter this effect, as they both increase the probability of good solutions to leave progeny in the next generation. When dealing with more complex and rugged fitness landscapes, however, other factors come into play. It is well known that added noise helps avoid trapping in local extrema, an effect used in optimization techniques such as simulated annealing. This effect can in some cases lead to an overall positive effect of noise, despite the obvious negative effects discussed above. Bäck and Hammel [2] note that for the generalized function by Ackley, the addition of some evaluation noise is beneficial to the convergence reliability of Evolution Strategies in a noisy environment. As we shall see, this phenomenon is not unique, and can be observed in other cases as well.

To check the effect of noise on the evolution of agents, we compared the mean population fitness and the fitness of the best individual in the last generation of 4 runs that used re-sampling factors of 50 to 200, to these figures in 20 runs with no re-sampling. The re-sampled runs obtained significantly inferior results in both the average last generation population fitness (25% mean difference, $p < 0.01$) and the maximum last generation fitness (24% mean difference, $p < 0.05$). These results are summarized in table 2. The P_θ measure used in the rest of the article was impractical in this case, since each run with a re-sampling factor of 200 lasted several weeks.

Table 2: Summary of results for the agent evolution problem. There were 20 runs with no re-sampling, and 4 runs with re-sampling factors of 50 to 200.

	Re-sampled runs	Non-re-sampled runs
Average last generation fitness	9.1725 ± 2.7271	6.8658 ± 4.0884
Maximum last generation fitness	12.3908 ± 3.5096	9.3129 ± 4.7900

Figure 1A shows the effect of noise on performance of the Standard GA for function f_4 , using the P_θ measure. Figure 1B plots the fitness of the best solution in the population as a function of the generation, for each of the four noise levels checked (averaged over 1000 distinct simulations in each case). While the averaged optimal fitness plot does not give clear indication about the effect of noise, it can clearly be seen from the P_θ plot that f_4 indeed benefits from the addition of noise. Even if it does not affect the average value of the best solution (across different runs), the addition of noise increases the variance of this value.

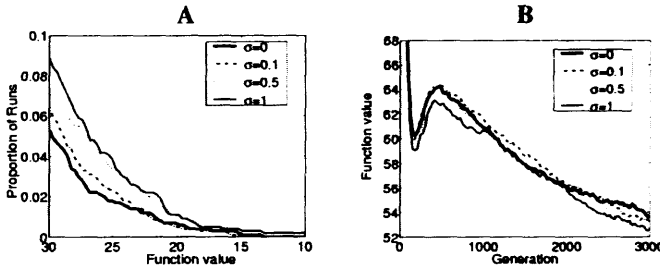


Figure 1: The effect of noisy evaluation on the performance of the Simple GA, for function f_4 . Results shown are based on 1000 independent optimization runs. A. P_θ measure. For each function value (with lower values representing a closer proximity to the optimum), the figure plots the proportion of runs in which at least one solution achieved that value or better (lower). Thus, a higher P_θ curve means better performance. B. Population best solution, averaged over the 1000 trials (lower value is better). Note the kink around generation 200, representing a reproducible escape from a strong local optimum.

Consequently, added noise can also increase the frequency of very good solutions, in which we are interested.

4.1 A Simple Analytical Model

A simple two-loci two peaks model can provide some insight into the positive effects of evaluation noise. This model assumes a population in which there is a common type ab , with fitness μ_1 , and two single-mutants, Ab and aB , with a lower fitness $\mu_2 < \mu_1$. We are interested in the expected time till the appearance of a superior double-mutant AB , which has fitness $\mu_3 > \mu_1$. We shall assume truncation selection, with a Gaussian evaluation noise of variance σ^2 around the actual fitness of each individual.

Denote by f_{ab} , f_{Ab} , and f_{aB} the frequencies of the common type and of the two single mutants, in a population with no double mutants. The survival probability of an individual is proportional to the probability of its measured fitness to pass a certain threshold $\theta = \mu_1 + K_1 \sigma$. Let K_2 be $(\theta - \mu_2)/\sigma$, and define P_1 and P_2 by $P_i = P(X_i > \theta) = P(Z > K_i)$ ($i = 1, 2$), where X_i is a normal random variable with expectation μ_i and variance σ^2 , and Z is a standard normal random variable. Then the survival probabilities of the common type and the single mutants are given by

$$\phi_i = \frac{P_i}{P_1 f_{ab} + P_2 (f_{Ab} + f_{aB})} \quad (i = 1, 2) \quad (1)$$

In a model with mutation rate m for each locus, and no recombination, the expected frequency of the double mutant in the next generation would be given by

$$\hat{f}_{AB} = m(1 - m)\phi_2(f_{Ab} + f_{aB}) + m^2\phi_1 f_{ab} \quad (2)$$

Substituting $\phi_1 = [1 - \phi_2(f_{Ab} + f_{aB})]/f_{ab}$ in equation (2) yields

$$\hat{f}_{AB} = m^2 + \phi_2(m - 2m^2)(f_{Ab} + f_{aB}) \quad (3)$$

which is monotonically increasing with ϕ_2 for any $m < 1/2$. Since ϕ_2 itself is monotonously increasing with σ , we can infer that adding noise to the measurement of the fitness indeed shortens the expected time till the appearance of the superior double mutant. A similar result can be obtained in a model with both mutation and recombination.

While in many cases reducing the evaluation noise is indeed beneficial, we see that this is not a general truth. In some cases, and in particular in some realistic scenarios in which the computational cost is especially high, reducing the noise not only does not help, but it actually significantly decreases performance. In the following, we shall see that even when noise-reduction has positive effects, these effects are negligible compared to the effects of using elitist selection to the same end.

5 Alternatives to Noise Reduction

5.1 Lessons from Nature

To some extent, evaluation noise can be considered as a variation generating force. Though not directly creating variation, noise allows currently deleterious variation to survive and possibly create beneficial combinations in the future. Let us therefore consider how variation affects evolutionary dynamics. The main variation-generating operators, mutation and crossover, have conflicting effects: Increased rates of mutation and crossover yield more variants, thus shortening the expected time until a superior mutant is discovered within an arbitrary population. On the other hand, increased variation rates tend to destroy beneficial combinations more quickly, thus reducing the chance of such a superior mutant to establish itself in the population.

The process of natural selection is inherently a very noisy process. It thus makes sense to examine the evolutionary mechanisms that can be found in nature itself. A particularly interesting model for studying the effects of variation on evolutionary dynamics is provided by facultative sexual organisms such as yeast. These organisms can reproduce either vegetatively, producing a copy of themselves which is identical to the original up to mutations, or sexually, by performing recombination with a second organism. It turns out that the tendency of such organisms to reproduce sexually is increased when they are subject to stressful conditions such as hunger, excess heat, irradiation, etc. [4, 3]. This reaction may be a special case of a wider phenomenon, which seems to exist in obligatory sexual organisms as well: recombination rates are often negatively correlated with the fitness of the organism. This has been observed in many different organisms, including bacteria, yeast and drosophila [18, 14, 22]. We further know that in bacteria, mutation rates are also negatively correlated with fitness, via the SOS mechanism. Bacteria experiencing stress tend to move into a hyper-mutation state, thus allowing more variation [19, 15].

In a couple of biologically-oriented works [10, 9] we have studied the phenomenon of Fitness-Associated Recombination (FAR), showing that it is beneficial from an evolutionary perspective both for the individual and for the population as a whole: Organisms which employ such a FAR strategy are likely to have more progeny in the long run. Populations in which recombination is fitness-associated tend to have a higher average fitness than populations in which individual recombination rates are independent of fitness. The latter is most pronounced when adaptation to a new environment is considered, and especially when dealing with a rugged adaptive landscape. The basic intuition behind this effect is simple: negative correlation between recombination and fitness reduces the destruction rate of good combinations, while maintaining their creation rate.

5.2 Elitism as Fitness-Associated Variation

Trying to apply the insights gained from nature to evolutionary computation, this simple principle can be taken to the extreme. In evolutionary computation techniques, one can *ensure* that the solutions with the highest potential pass unchanged to the next generation, rather

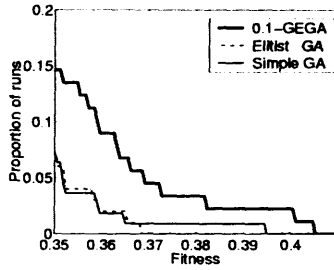


Figure 2: The effect of group-elitist selection in a real-world agent evolution problem. The plotted P_θ curves indicate, for each fitness value (X axis), the proportion of runs (Y axis) in which this performance level or better was achieved by at least one individual. Note that contrary to the synthetic benchmarks, here higher values along the X-axis mean better performance. A higher P_θ curve indicates better overall performance of the algorithm.

than just increase their survival and reproduction chances. This takes us from a principle which effects the distribution of *variation*, to one that also alters the way *selection* acts on the population. Indeed, the well established notion of *Elitist Selection* can be seen as an extreme form of fitness-associated variation. Forms of elitism include the Simple Elitist Genetic Algorithm (SEGA) [12], $(\mu + \lambda)$ -Evolution Strategies [20], Elitist Recombination [21] and other, less well-known, selection schemes. The common feature among all these selection schemes is that they preserve the best solution in the population intact, or replace it with even better solutions.

Elitist selection schemes have received attention due to their analytically tractable convergence properties and the general usefulness of ensuring a non-decreasing population optimum. Yet the importance of elitism in the context of noisy objective functions seems to have been overlooked, in favor of population sizing and fitness re-sampling considerations. We will now focus on the particular benefits that elitism can have in the context of noisy evaluation.

The Group-elitist Genetic Algorithm (GEGA) is a straightforward generalization of the Simple GA, in which a fixed-size subset of the parent population is passed into the offspring population. Put in other terms, GEGA is a simple GA with generation gap smaller than 1 and fitness-based parent re-insertion. We shall denote by γ -GEGA ($\frac{1}{N} \leq \gamma < 1$, where N is the population size) the GEGA algorithm with an elite group of size γN . The cases $\gamma = \frac{1}{N}$ (SEGA) and $\gamma = \frac{N-1}{N}$ (Steady-State GA) have received special treatment in the past [13]. We shall focus our attention on the case $\frac{1}{N} < \gamma \ll 1$.

While SEGA is often very useful in noiseless conditions, its effectiveness is greatly reduced in the presence of noise, since it often preserves the wrong solution. By maintaining a small elite *group*, one can avoid the harmful effects of evaluation noise, without losing its benefits. The optimal size of the elite group depends on the fitness landscape, and should be positively correlated to the noise level. Since we are mainly interested in cases where the fitness landscape is non-trivial, and its properties are often unknown, it is impractical to calculate or find in advance the optimal elite size for each problem. Unless otherwise stated, we will therefore use $\gamma = 0.1$ as a reasonable compromise. This setting has resulted in improved performance for all the cases we studied so far.

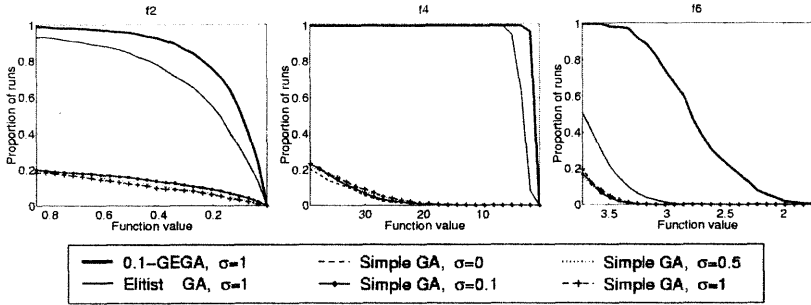


Figure 3: The effects of noise reduction and of elitist selection on the reliability of obtaining the best results. P_θ measures are shown for 3 benchmark functions. For each function value, the curves show the proportion of runs in which at least one solution achieved that value or better (lower). Thus, a higher P_θ curve means better performance of the respective algorithm. Results are based on 1000 independent optimization runs.

5.3 Simulation results

Figure 2 shows the effect of Group-Elitist selection on the evolution of Neuro-controlled Autonomous Agents. This data was obtained from over 650 very long runs, lasting 10000 generations. The P_θ measure clearly demonstrates the superiority of 0.1-GEGA over a Simple Elitist GA and a Simple GA for obtaining top-level solutions with high reliability. Note that in this case the objective was to maximize the fitness, thus higher θ values mean better performance, contrary to the case of the benchmark functions.

The effect of the elitist procedure was overwhelming and fully consistent for all the benchmark functions we checked. Reducing the noise had at best a negligible effect compared to that of group-elitist selection. Four levels of noise were examined for each function, $\sigma = 1, 0.5, 0.1, 0$. Taking $\sigma = 1$ as the basic noise-level, achieving the lower noise levels corresponds to re-sampling factors of 4, 10, and ∞ , respectively. Taking other γ values between 0.05 and 0.2 resulted in performance which was almost indistinguishable from that of $\gamma = 0.1$.

Figure 3 shows the P_θ performance for three of the benchmark functions, under six conditions: the four noise levels, the basic noise level $\sigma = 1$ with a Simple Elitist Algorithm, and the basic noise level with 0.1-GEGA. The X axis range $[\theta_1, \theta_2]$ was chosen so that $P(\theta_1) = 0.2$ for the noiseless Simple GA, whereas $P(\theta_2) = 0.005$ for the 0.1-GEGA case, ensuring coverage of our “area of interest”. Both elitist schemes are much better than any of the naive cases, with or without noise reduction. In fact the differences in performance between the different noise levels are negligible in comparison to the improvement caused by using the Group-Elitist scheme. Similar results were obtained for the three other benchmark functions as well. 0.1-GEGA consistently achieved better performance than simple elitism. This result was most pronounced for f_6 - a rugged function of high dimension.

6 Discussion

Elitist selection and noisy fitness evaluation have both received attention in the literature. Yet the applicability of elitism to the case of noisy fitness landscapes has been largely overlooked. While noisy evaluation has harmful effects on the convergence properties of Evolutionary Algorithms, we showed that its overall effect is not always negative. We suggest that this is due to a ‘warming’ effect of the evaluation noise, reducing the chance of trapping in local fitness optima.

Motivated by biological insights, we suggested fitness-associated variation principles as

an alternative solution to the problems associated with evaluation noise. In the realm of Evolutionary Computation, elitist selection can be regarded as an extreme and very simplified form of the same principle. We have shown that applying elitist selection schemes has a far greater effect on overall GA performance than the amount of evaluation noise in the absence of elitism. Group-Elitism, preserving a certain percentage of the best solutions in the population, is an elitist selection scheme particularly suitable for the case of noisy evaluation. We have demonstrated the effectiveness of this technique on real-world problems, as well as a variety of synthetic benchmark functions. Contrary to re-sampling or resizing, this procedure has a negligible computational cost, and it leads to superior performance in non-trivial cases.

The optimal value of γ , the Elite Group percentage, depends on the properties of the fitness landscape, as well as on the noise level in the evaluation process. Since these properties are usually not known in advance and are very costly (if not altogether impossible) to estimate, the optimal γ is hard to calculate. Fortunately, the qualitative effect seems to be quite stable to changes in the exact value of γ , and choosing $\gamma = 0.1$ proved to be a good rule of thumb, resulting in excellent performance in all the cases studied so far. Comparison to re-sampling and population resizing, the two standard procedures applied to evolutionary techniques with noisy fitness landscapes, GEGA enjoys better performance and is virtually cost-free. Understanding the relation between the fitness landscape and the optimal value of γ will be an interesting subject for future research.

Acknowledgments: This research was supported in part by the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities. We are grateful to Eytan Ruppin, Ilan Eshel and Henri Atlan for many stimulating discussions and valuable comments. We thank Ranit Aharonov-Barki for reading the manuscript and providing helpful remarks.

References

- [1] R. Aharonov-Barki, T. Beker, and E. Ruppin. Emergence of memory-driven command neurons in evolved artificial agents. *Neural Comp.*, 13(3), March 2001.
- [2] T. Bäck and U. Hammel. Evolution strategies applied to perturbed objective functions. In *Proc. IEEE World Cong. Comp. Intel.*, pages 40–45, 1994.
- [3] G. Bell. *The Masterpiece of Nature: The Evolution and Genetics of Sexuality*. Croom Helm, 1982.
- [4] L. Davis and R. Smith. Meiotic recombination and chromosome segregation in *Schizosaccharomyces pombe*. *Proc. Nat. Acad. Sci. USA*, 98:8395–8402, 2001.
- [5] J. Fitzpatrick and J. Grefenstette. Genetic algorithms in noisy environments. *Machine Learning*, 3(2/3):101–120, 1988.
- [6] T. Fogarty. Reproduction, ranking, replacement and noisy evaluations: experimental results. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, page 634. Morgan Kaufman, 1993.
- [7] D. Fogel. *Evolutionary Computation*. IEEE Press, 1995.
- [8] D. Goldberg, K. Deb, and J. Clark. Genetic algorithms, noise and the sizing of populations. *Complex Syst.*, (6):333–362, 1992.
- [9] L. Hadany and T. Beker. Fitness-associated recombination on rugged adaptive landscapes. In submission, 2002.
- [10] L. Hadany and T. Beker. On the evolutionary advantage of fitness-associated recombination. In submission, 2002.
- [11] U. Hammel and T. Bäck. Evolution strategies on noisy functions: How to improve convergence. In Y. Davidor, H.-P. Schwefel, and R. Manner, editors, *Proc. Int. Conf. Evol. Comp., Jerusalem*, pages 159–168, 1994.

- [12] K. D. Jong. *Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Ann Arbor, 1975.
- [13] K. D. Jong and J. Sarma. Generation gaps revisited. In L. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 19–28. Morgan Kaufmann, San Mateo, CA, 1993.
- [14] A. B. Korol, L. A. Preygel, and S. I. Preygel. *Recombination Variability and Evolution*, chapter 6 & 9. Chapman & Hall, London, 1994.
- [15] G. McKenzie, R. Harris, P. Lee, and S. Rosenberg. The SOS response regulates adaptive mutation. *Proc. Nat. Acad. Sci. USA*, 97:6646–6651, 2000.
- [16] B. Miller and D. Goldberg. Genetic algorithms, selection schemes, and the varying effects of noise. *Evol. Comp.*, 4(2):113–132, 1996.
- [17] M. Rattray and J. Shapiro. Noisy fitness evaluation in genetic algorithms and the dynamics of learning. In M. D. V. R. Belew, editor, *Foundations of Genetic Algorithms IV*, pages 117–139. Morgan Kaufmann, 1997.
- [18] R. Redfield. Do bacteria have sex? *Nat. Rev. Gen.*, 2:634–639, 2001.
- [19] S. Rosenberg. Evolving responsively: Adaptive mutation. *Nat. Rev. Gen.*, 2:504–515, 2001.
- [20] H. Schwefel. *Numerical Optimization of Computer Models*. John Wiley, 1981.
- [21] D. Thierens. Selection schemes, elitist recombination and selection intensity. In *Proc. 7th Int. Conf. Gen. Alg., ICGA-97*, pages 152–159, 1997.
- [22] N. Tucić, F. J. Ayala, and D. Marinković. Correlation between recombination frequency and fitness in *Drosophila melanogaster*. *Genetica*, 56:61–69, 1981.

This page intentionally left blank

Section 5

Machine Learning

This page intentionally left blank

Anticipatory Computing with Autopoietic and (M, R) System

Juan Carlos Letelier, Gonzalo Marín, Jorge Mpodozis and Jorge Soto-Andrade

Departamento de Biología, Facultad de Ciencias, Universidad de Chile

1. Introduction

From the many attempts to produce a conceptual framework for the organization of living systems, the notions of (M,R) systems (Rosen, 1958; Rosen, 1959) and Autopoiesis (Maturana, 1970; Maturana and Varela, 1972; Maturana and Varela, 1980) stand out for their rigor, their presupposition of the circularity of metabolism, and the new epistemologies that they imply. From their inceptions, these two notions have been essentially disconnected because each has defined its own language and tools. Here we demonstrate the existence of a deep conceptual link between (M,R) systems and Autopoietic systems. This relationship permits us to posit that Autopoietic systems, which have been advanced as capturing the central aspects of living systems, are a subset of (M,R) systems. This result, in conjunction with previous theorems proved by Rosen, can be used to outline a demonstration that the operation of Autopoietic systems cannot be simulated by Turing machines. This demonstration is rather delicate as it involves four different aspects: a) “the central theorem of Rosen” involving the stability of metabolic networks with circular organization, b) the use of the theory of Categories to model systems with circular organization, c) a preliminary proof that such systems can not be computable “*a la Turing*” and d) an identification between Autopoietic and (M,R) systems. This work will analyze with detail the mathematics behind Rosen central theorem as well as the implications for computing of the intrinsic anticipatory behavior of Autopoietic systems. The unfamiliar computing aspects of Autopoietic system arise from the internal

reference frame of the controller. The control is done by the logic of the maintenance of circular organization [Caspi, 2002] in the presence of structural coupling.

2. Discussion

Autopoietic and (M,R) systems define the problem of circular organization as the core of living systems but approach this circularity from two different perspectives. While Rosen tried, with (M,R) systems, to prove that circular organization can arise spontaneously from the functions of metabolism and repair inside a metabolic network with a time-invariant organization, Autopoiesis explored the biological consequences of a living system characterized by circular organization acting (behaving or interacting) with its environment. (M,R) systems builds a theory to describe and manipulate processes, in general, and metabolic networks, in specific, along with how to understand or deduce, metabolic time invariance (Rosen, 1985). Autopoiesis explores the consequences of a system's operation with circular organization that separates itself from its surroundings. Both theories posit that the core of biological phenomena arises from circular organization, and not from information processing, reproduction, the generation of "the" correct response to an outside stimulus or optimizing metabolic fluxes by minimizing energy use.

The idea of using the formalism of (M,R) systems as a possible framework of Autopoietic systems was previously advanced in the context of *Quasi-Autopoietic systems* (QAP) (Nomura 1997). Despite their name, QAP systems are far more similar to (M,R) systems than to Autopoietic systems. Instead of having a replication map \square QAPs have an iteration map that defines the internal dynamics through a recursive procedure. QAPs also lack a specific metabolism and, more importantly, they do not consider the existence of the boundary. The mathematical efforts behind QAPs are interesting but, in our opinion, they fail to grasp the fundamental biological relationship between (M,R) and Autopoietic systems. The theory of categories has also been used to model the internal organization of a living system, its evolution and its levels of organization (Ehresmann and Vanbremeersch 1987). This model, known as *Systemes Evolutifs avec Memoire* (SEM), is a technically complex formalization of biological systems that crucially ignores operational closure or circular organization and, in this sense, is a theoretical effort centered on a categorical representation of living systems but very different from (M,R) systems.

By virtue of Autopoietic systems' essential turnover of components, as well as the destruction and creation of whole classes of molecules during ontogeny, these systems cannot be characterized within the scope of traditional Dynamical Systems Theory. As their structure can change, without changing the organization, Autopoietic systems cannot be described with a fixed-state space (Kampis 1991). The challenge is to use the categorical formalism of (M,R) systems to develop a categorical representation of Autopoietic systems. This specific work should be focused on finding a categorical representation of a system that is circularly closed and produces its own boundary. In summary the categorical representation used by Rosen (1972) must be refined to include the (extra) properties that make an (M,R) system an Autopoietic system. The relevance of the system's boundary revealed by our theoretical analysis is currently paralleled by the experimental efforts to synthesize proto-cells and the importance that metabolic compartmentalization acquires in such experiments (Szostak, Bartel et al. 2001). Without any doubt the central theorem of Rosen must be understood in its proper context. All demonstrations rely in the (amazing and unusual) property of the existence of invertible evaluation functions.

Autopoietic systems' non-computability by Turing Machines has many important theoretical consequences. First, it limits the validity of mimesis (i.e. simulation) as a means to understand living systems. In effect, this result shows that the phenomenology that arises from the circularity of metabolism cannot be simulated with current computer architectures based on the Von-Neumann implementation of Turing machines. Using different approaches this result has been hinted at on at least two occasions in the last decade. Using formal arguments, Boden argued for the impossibility of designing a living system without a real metabolism, thereby raising serious doubts about the conceptual program of Strong Artificial-Life (Boden 1999). On the other hand, Kampis has developed a concept of living systems, which he calls "Component-systems," and he shows that equations of state, equations of motion or evolution equations cannot be applied to Component-systems (Kampis 1991). The non-computability of Autopoietic systems, as advanced here, apparently collides with the simulation results involving tessellation automatas (Varela, Maturana et al. 1974). But new versions of this simulation show that the original report of computational Autopoiesis was flawed, as it used a non-documented feature involving chain-based bond inhibition (McMullin and Varela 1997). Thus the

closure exhibited by tessellation automatas is not a consequence of the “network” of simulated processes, but rather an artifact of coding procedures. Thus our point concerning the non-computability of Autopoietic systems appears supported by the more modern simulations. In any case, as our analysis has shown, the failure of closure in these computational models cannot be construed, in any way, as a conceptual failure of Autopoiesis, instead it reflects the non-computability of Autopoietic systems.

The non-computability of Autopoietic systems could initially appear as an incredibly (or suspiciously) strong result, but even in the restricted field of pure Mathematics it has been possible to prove the existence of simple, but non-computable functions like the *busy beaver* problem (Rado 1962). Thus Turing non-computability is a property that does not require the complexities of circular organization to be apparent, as it is already demonstrable in simpler systems or problems. The failure of the Turing-Church hypothesis with respect to Autopoietic systems opens some important new questions. The first challenge would be to analyze whether an Autopoietic system can implement a Turing machine. The second, and far more interesting question, is to consider whether some Turing non-computable problems, like the busy beaver, can be computed by Autopoietic systems. These considerations belong to the new field of Emergent Computation or Bio-Computing. To tackle these problems, it is essential to expand the tools developed by Rosen, essentially the use of the theory of categories to represent Autopoietic systems and to understand and manipulate the operational closure of metabolism.

Autopoietic systems do not simply behave or exist passively in an environment. A central aspect of Autopoiesis is the idea of *structural coupling*, a mechanism by which the living system and its environment determine, in a mutual way, some of their properties. This idea could be the basis of a new type of biologically-oriented computation, which would not be program based. Because Autopoietic systems do not have inputs or outputs, only a circular dynamic which is perturbed but not defined by external agents, it is not possible to encode outside concepts into Autopoietic states, nor to control a trajectory of states (like Turing machines). Thus an external observer can only define a computation for an Autopoietic system as the particular ontogeny for that system. During the system’s ontogeny, a relation between it and its medium is *selected* or *stabilized*. This relation has meaning, in the sense of the *Umwelt*, for the Autopoietic system, which is structurally coupled to its medium, but

not for external observers. Thus external observers, if they wish to use Autopoietic systems to perform computations, must find a procedure to attach meaning to particular moments and properties of the system's ontogeny (Letelier, Marín and Mpodozis, 2002).

Considering Autopoietic systems as a subset of (M,R) systems raises questions such as: a) how to map the concept of Φ onto the formalism of Autopoietic systems; b) how the notions of structure and organization of an Autopoietic system map onto (M,R) systems. Furthermore, Φ is a conflicting notion, as it is a functional component that has the function of selecting one realization of metabolism among many possible instances. Φ is a functional component (i.e. a relational entity) that does not map 1-1 onto physical entities. As Autopoiesis begins from the notion of circular organization (it does not try to prove how it can arise), Φ appears, in a first approximation, as an alien concept to Autopoiesis. But a consideration of the very definition of an Autopoietic system ("...[an Autopoietic system] is a network of processes configured in a....") suggests a possible identification for Φ . From the point of view of Autopoiesis, Φ could be thought of as the configuration of processes that establishes a circular, time-invariant network. It is exactly this configuration that moment by moment drives the metabolism along the lines of circularity and stability. Thus a profound identification can be found between Φ and the "metabolic network or mesh" that defines an Autopoietic system. The clarification of this link is, without a doubt, one of the issues that must be addressed first.

The conclusion that Autopoietic systems are a subset of (M,R) systems could be rather surprising for some. As Autopoiesis is the direct offspring of a very general viewpoint of living systems, where the nature of the act of observing is the fundamental step, its viewpoint seems more general than the objectivistic vantage of early system analysis, which is the (implicit) foundation of (M,R) systems. Thus it could appear that (M,R) systems are more specific, and less general, than Autopoietic systems. We claim that, in this respect, it is important to distinguish between the *epistemological frameworks* of the models developed by Rosen, on one hand, and Maturana and Varela, on the other, and the *systems* (Autopoietic or (M,R)) that these two epistemologies bring forth. In this paper we avoided the interesting point of comparing these epistemologies. Instead we focused on the more restricted domain of how these two classes of systems are related. Because Autopoietic systems incorporate, in the notion of separation from the environment, the

discrete nature of living systems, we conclude that Autopoietic systems, although less general than (M,R) systems, capture the essential points of living organization as they couple circularity to discreteness. Thus we have a dual situation in which *the epistemological framework* of Autopoiesis is more general than (M,R) systems (for example only one type of object exists in Autopoietic systems (called components) while transformable materials, components and repair subsystems compose (M,R) systems), but operationally an (M,R) system is more general than an Autopoietic system as it has only the property of circular cellular organization and lacks spatial confinement.

We consider that our principal contribution is connecting the two, until now, disconnected theories and using this link to prove that Autopoietic systems are not Turing-computable. The Autopoietic model has gained the possibility of using the theory of categories to describe processes and networks of processes. On the other hand (M,R) systems, along with QAPs and *SEM*, can benefit or be complemented by the more biologically oriented *Weltanschauung* of Autopoiesis revealed through its application to problems like the origin of life (Mavelli and Luisi 1996), evolution (Maturana and Mpodozis 2000) or neurobiology (Mpodozis, Letelier et al. 1995). The demonstration of Autopoiesis' inclusion in (M,R) systems presented here is only the first step in a future synthesis. Besides the necessity of establishing a map between Φ and Autopoietic concepts, other very difficult points need to be addressed, including how to incorporate the notion of the observer in defining "objects" that are not independent from the observing act.

Acknowledgments. We thank Dr. Humberto Maturana for his continuous support and understanding. (Supported by Fondecyt 1990045 to J.M.).

References

- [1] Boden, M. (1999). "Is Metabolism Necessary?" *Brit. J. Philo. Sci.* **50**(2): 231-249.
- [2] Caspi, J. (2002) "Biologizing" control theory: How to make a control system come alive. *Complexity* **7**(4), 10-13.
- [3] Ehresmann, A. C. and J. P. Vanbreemersch (1987). "Hierarchical evolutive systems." *Bull. Math. Biol.* **49**(1): 15-50.
- [4] Kampis, G. (1991). Self-modifying systems in biology and cognitive science. Oxford, Pergamon Press.

- [5] Letelier, J.C., Marín, G. and Mpodozis, J. Computing with Autopoietic Systems. in *Soft Computing and Industry: Recent Applications*. (Eds). Roy, R., Koppen, M., Ovaska, S., Furuhashi, T., and Hoffmann, F. Springer, London (ISBN:1852335394)
- [6] Maturana, H. (1970). The neurophysiology of cognition. Cognition, a multiple view. P. Garvin. New York, Spartan Books: 3-23.
- [7] Maturana, H. and J. Mpodozis (2000). "The origin of species by means of natural drift." Rev. Chil. Hist. Nat. **73**: 261-310.
- [8] Maturana, H. and F. Varela (1972). De Máquinas y Seres vivos. Santiago, Editorial Universitaria, Universidad de Chile.
- [9] Maturana, H. and F. Varela (1975). Autopoietic System. A characterization of the living organization. Urbana, Biological Computer Laboratory.
- [10] Maturana, H. and F. Varela (1980). Autopoiesis and Cognition: The realization of the living. Dordrecht, Reidel.
- [11] Mavelli, F. and P. Luisi (1996). "Autopoietic self-reproducing vesicles: a simplified kinetic model." J. Phys. Chem. **100**: 16600-16607.
- [12] McMullin, B. and F. Varela (1997). Rediscovering computational autopoiesis. European Conference Artificial Life, Brighton.
- [13] Mpodozis, J., J. C. Letelier, et al. (1995). "Nervous system as a closed neuronal network: Behavioral and cognitive consequences." Lecture Notes in Computer Science. **930**: 130-136.
- [14] Nomura, T. (1997). An attempt for description of quasi-autopoietic systems using metabolism-repair systems. Fourth European Conference on Artificial Life, MIT Press.
- [15] Rado, T. (1962). "On Non-computable functions." Bell Sys. Tech. J. **41**(3).
- [16] Rosen, R. (1958). "The Representation of biological systems from the standpoint of the theory of categories." Bull. Math. Biophys **20**: 317-341.
- [17] Rosen, R. (1959). "A relational theory of biological systems II" Bull. Math. Biophys. **21**: 109-128.
- [18] Rosen, R. (1964). "Abstract biological systems as sequential machines II: Strong connectedness and reversibility." Bull. Math. Biophys. **26**: 239-246.
- [19] Rosen, R. (1966). "Abstract biological systems as sequential machines: III. Some algebraic aspects." Bull. Math. Biophys. **28**: 141-148.
- [20] Rosen, R. (1985). Anticipatory Systems. Oxford, Pergamon.
- [21] Rosen, R. (1991). Life itself. New York, Columbia University Press.
- [22] Rosen, R. (2000). Essays on life itself. New York, Columbia University Press.
- [23] Szostak, J. W., D. P. Bartel, et al. (2001). "Synthesizing life." Nature **409**(6918): 387-390.
- [24] Varela, F., H. Maturana, et al. (1974). "Autopoiesis: The organization of living systems, its characterization and a model." Biosystems. **5**(4): 187-196.

An Efficient Algorithm in Optimal Partition Problem for Trees Induction

Mounir Asseraf
ESIEA Recherche
Parc Universitaire Laval-Changé
38, Rue des Docteurs Clamette et Guérin-53000 LAVAL
Tel : 02 43 59 46 19
E-mail : asseraf@esiea-ouest.fr

Abstract

Addressing the inherent computational complexity of the construction of optimal trees, we will present in this paper an efficient procedure to find the optimal partition for categorical variables. The attribute selection metric will be presented for this optimisation. It's the Kolmogorov-Smirnov criterion adapted to discrete variables.

The algorithm converges to the globally optimal solution in polynomial time with three degrees.

We will compare the complexity time with other classical algorithms and show that there is a significant difference in time required to find the optimal partition.

1. Introduction

In classification trees, the model is to construct a binary tree for predicting the class to which an object belongs on the basis of vector of measurements of that object. Classification trees are also called decision trees in the literature and have been well studied, with applications in such areas as pattern recognition [8 ; 13], logic design [2], taxonomy, questionnaires, and diagnostic manuals [9], expert systems, and machine learning [10].

The data from which the rule will be constructed comprises the measurement vectors and class identifiers for a sample of objects taken from the same distribution as the objects to be classified.

The success of these methods lies in the facility of interpretation of the results and simplifies as the rule to be applied and interpreted by someone who does not understand the underlying statistical and computational methodology (e.g. medicine, retail banking and marketing).

This paper describes the attribute selection metric which will be adapted for categorical variables. It is conceptually straightforward, so it is easy to be interpreted. This metric is Kolmogorov-Smirnov (ks) adapted for qualitative variables, it can straightforwardly handle many of the characteristics commonly found in the large datasets found in the emerging field of data mining. Moreover, considering the discriminating capacity satisfying this distance, it can be used for the two types of predictors, which is frequent in the data mining

applications –many thousands of instances and feature vectors contain both numerical and categorical predictors-.

We will also describe the speed algorithm for finding the optimal partition to split the population in each node to two subpopulations.

Formally, models in classification trees provide estimates of the conditional probabilities $p(Y / X)$, where Y represents the prior classes and taking K discrete values, C_1, C_2, \dots, C_K and X is a set of P predictors.

In the bayesian approach, Friedman [7] supposes that the cost of misclassification and the probability of the prior class $\pi_k (k = 1, \dots, K)$ are inversely proportional. Under this hypothesis we prove that the search of the minimum of the Bayes' risk is equivalent to the maximum of ks distance.

Explicitly, suppose that X is real and let be F_1, F_2 the cumulative distribution respectively associated to prior classes C_1, C_2 , we have to find a threshold x^* such that :

$$ks(x^*) = \max_x |F_1(x) - F_2(x)| \quad (1)$$

When there are more than two classes, [3] proposed to construct a tree for these classes, using the twoing-method : two super-classes are built by splitting a set $\{C_1, \dots, C_K\}$ onto two groups A_1, A_2 such that $A_1 \cap A_2 = \emptyset$ and $A_1 \cup A_2 = \{C_1, \dots, C_K\}$.

Hence, the Kolmogorov-Smirnov distance is defined by

$$ks(F_{A_1}(x), F_{A_2}(x)) = |F_{A_1}(x) - F_{A_2}(x)| \quad (2)$$

where

$$F_{A_k}(x) = \frac{1}{\pi_{A_k}} \sum_{i \in A_k} \pi_i F_i(x)$$

$$\text{with } \pi_{A_k} = \sum_{G_i \in A_k} \pi_i.$$

The recursive partitioning is similar to the binary case ($K=2$).

We adapt the Kolmogorov-Smirnov distance when X is a discrete variable [1] as follows: the cumulative distribution associated to each class ($k=1,2$) is

$$F_k(x) = \frac{P(C_k \cap [X = x])}{\pi_k} = P(x / C_k) \quad (3)$$

and then we define $ks(F_1(x), F_2(x)) = |F_1(x) - F_2(x)|$

As in the case when X is real, the set x^* that minimizes the Bayes' risk of misclassification is the set that maximizes the quantity

$$ks(F_1(x^*), F_2(x^*)) = \max_x ks(F_1(x^*), F_2(x^*)) \quad (4)$$

In the multivariate case $X = (X_1, \dots, X_p)$ the optimal partitioning is obtained by selecting the subset x_j^* such that

$$ks(F_1(x_j^*), F_2(x_j^*)) = \max_j ks(F_1(x_j), F_2(x_j)) \quad (5)$$

where $x_j^* = \arg \max_{x_j} ks(F_1(x_j), F_2(x_j))$.

When there are several classes ($g > 2$), the Kolmogorov-Smirnov distance is defined as follows :

if A_1, A_2 is a partition of C_1, C_2, \dots, C_K then

$$ks(F_{A_1}(x), F_{A_2}(x)) = |F_{A_1}(x) - F_{A_2}(x)| \quad (6)$$

where

$$F_{A_k}(x) = \frac{1}{\pi_{A_k}} \sum_{i \in A_k} \pi_i F_i(x)$$

with $\pi_{A_k} = \sum_{C_i \in A_k} \pi_i$.

2. Algorithm for the Optimisation of *ks* Adapted

Due to the inherent computational complexity of the construction of optimal trees, practical procedures for constructing them are almost universally steepest-descent greedy procedures which grow trees outward starting from the root.

We assume that X is a qualitative variable and not a vector of qualitative variables. This assumption doesn't affect the principle of the algorithm because the selecting cut of the optimal partition is always assigned for only one variable and not for the combination of variables.

We proved that for each subset x of categories and two super-classes A_1, A_2 , the optimal partition is obtained when we sort all the cumulative distributions $F_k(x)$, $k=1, \dots, K$. The smallest cumulative distribution in A_1 is larger than the biggest cumulative distribution in A_2 . Following that, the complexity time is reduced from exponential to linear, because for constructing all partitions of super-classes in range of K elements is equal to 2^K .

The intuitive content is clear. The best split should put in super-class 1 all those categories which have the highest probability to be in this class and those categories which have the lowest probability to be in the other super-class.

For example, let X be a qualitative variable which has 4 categories x_1, \dots, x_4 and Y a label of 6 prior classes C_1, \dots, C_6 . Suppose that a contingency table of 100 samples is given in Table1:

Table 1: Contingency table of (X,Y)

	x_1	x_2	x_3	x_4	Marg
C_1	3	3	4	2	12
C_2	11	8	6	5	30
C_3	3	4	5	2	14
C_4	6	6	3	5	20
C_5	7	3	2	1	13
C_6	6	2	2	1	11
Marg	36	26	22	16	100

We sort the cumulative distributions $F_k(x_i), k=1,...,6$ of the first column (x_1).

Table 2 gives the profile ordered according to the first column.

Table 2: Profile ordered by x_1

	x_1	x_2	x_3	x_4	Total
$F_6(x_i)$	0.545	0.181	0.181	0.09	1
$F_5(x_i)$	0.538	0.23	0.153	0.076	1
$F_2(x_i)$	0.366	0.266	0.2	0.166	1
$F_4(x_i)$	0.3	0.3	0.15	0.25	1
$F_1(x_i)$	0.25	0.25	0.333	0.166	1
$F_3(x_i)$	0.214	0.285	0.357	0.142	1

To find an optimal partition on $C_1,...,C_6$ associated to x_1 it suffices to evaluate each of the $(6-1=5)$ instead of $2^6 - 1 = 63$ possible thresholds, and to choose the partition with the best performance.

				ks
$A_1 = \{C_6\}$	against	$A_2 = \{C_1,...,C_5\}$		0.20837
$A_1 = \{C_6, C_5\}$	against	$A_2 = \{C_1,...,C_4\}$		0.23903
$A_1 = \{C_6, C_5, C_2\}$	against	$A_2 = \{C_1, C_3, C_4\}$		0.1835
$A_1 = \{C_6, C_5, C_2, C_4\}$	against	$A_2 = \{C_1, C_3\}$		0.17463
$A_1 = \{C_6, C_5, C_2, C_4, C_1\}$	against	$A_2 = \{C_3\}$		0.1694

The best performance in column x_1 is $(A_1^*, A_2^*) = (\{C_5, C_6\}, \{C_1, ..., C_4\})$ but it is not necessarily globally optimal because we must consider all the sets of possible partitions in space of X .

In spite of this reduction, the complexity is still exponential with respect to p .

Let A_1, A_2 be two super-class, X a qualitative variable with p categories and B_1 a nonempty subset of categories. There exists one category in B_1 that respects the same algorithm as described above.

The general algorithm is as follows :

We fix one category and we sort the K cumulative distributions and we calculate $K-1$ values of the ks -distance by separating, for the first level, the class having the greatest cumulative

distribution from the other classes. For the second level, the two classes having the largest cumulative distributions . And so on.

For each level (1 to $K-1$) we add linearly all the categories having given a positive ks without the absolute values.

The computational complexity of this algorithm is $p(p-1)(K-1)$, it is linear by K and quadratic by p .

In the above example the complexity of an exhaustive search through the collection of permissible tests is plotted in Figure 1

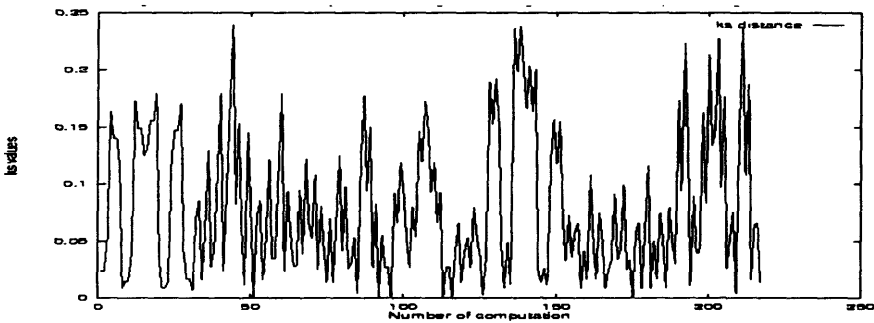


Figure 1: Search for the optimal partition by complete enumeration

Using our strategy the run time for searching for the optimal partition is cubic as a function of the parameters, this search is plotted in Figure 2.

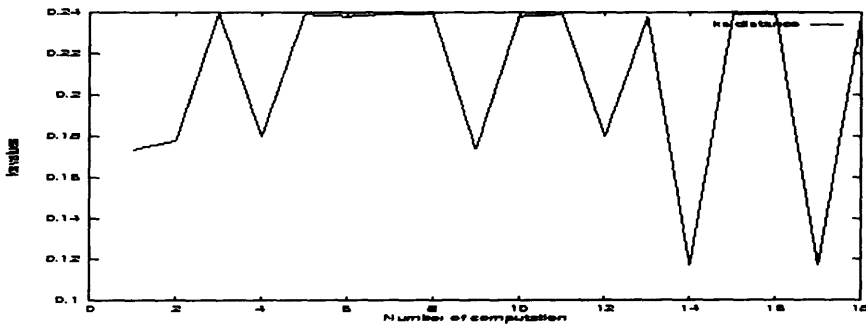


Figure 2: Our strategy search for the optimal partition

The X-axis represent the number of computations to calculate the ks distance, the Y-axis correspond to the value of the ks .

We can see in Figures 1 and 2 the same maximum value of ks distance but using our strategy there is a great gain in time.

3. Empirical Comparison with Classical Criteria

To evaluate the ks algorithm speed, we compared the test selection metric for *gini* and ks metrics. The performance differences that we measured are only due solely to the differences in these two algorithm metrics.

We chose the dataset waveform proposed by [3] to test this aspect of the problem.

In the waveform recognition problem, recall that the classes are superposition of two waveforms among three theoretical waveforms h_1, h_2, h_3 as sketched in Figure 3.

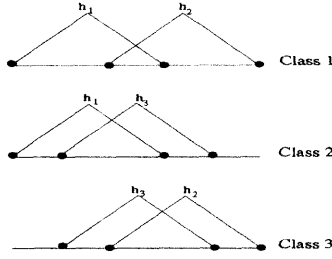


Figure 3: Waveforms

Each class consists of a random convex combination of two of these waveforms sampled at the integers with noise added. More specifically, the measurement vectors are 21 dimensional $X = (x_1, \dots, x_{21})$. To generate a class 1 vector, X independently generates a uniform random number u and 21 random numbers $\varepsilon_1, \dots, \varepsilon_{21}$ normally distributed with mean zero and variance 1. The procedure is the same for class 2 and class 3. Then set

$$\begin{aligned} \text{Class 1} \quad x_j &= uh_1 + (1-u)h_2(j) + \varepsilon_j & j &= 1, \dots, 21 \\ \text{Class 2} \quad x_j &= uh_1 + (1-u)h_3(j) + \varepsilon_j & j &= 1, \dots, 21 \\ \text{Class 3} \quad x_j &= uh_2 + (1-u)h_3(j) + \varepsilon_j & j &= 1, \dots, 21 \end{aligned}$$

we suppose that the prior classes are same probability (1/3, 1/3, 1/3). Figure 4 shows the waveforms when the noise is added.

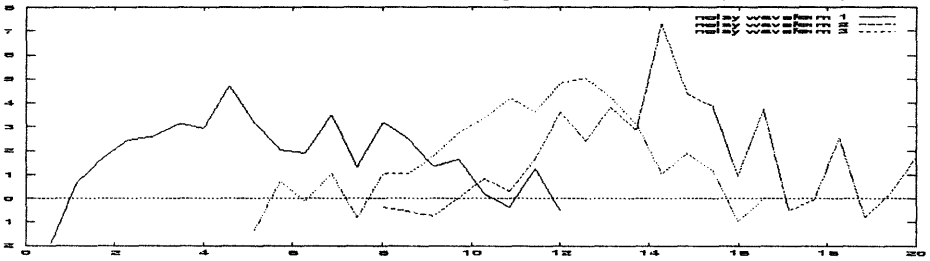


Figure 4 : noisy waveforms

We generated 5000 samples from these distributions, 301 constitute the learning sample and 4699 constitute the test sample.

We will compare our method (ks criterion for qualitative variables in multi-class) with Gini criterion proposed by [3] in a multi-class problem.

We have discretised each quantitative predictor x_1, \dots, x_{21} by inertia criterion on 2, 3, ..., 20 categories and then we generated $20-2+1 = 19$ datasets, denoted D1, D2, ..., D19.

We drew Table 3 to measure the execution time of each dataset of the two criteria. It is noticed that the execution time to build a decision tree by *ks* is definitely faster than that of *gini*.

Table 3 : execution time on second

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
<i>gini</i>	24	32	80	192	248	379	512	824	1433	1982
<i>ks</i>	22	26	32	40	50	62	76	92	110	130
	D11	D12	D13	D14	D15	D16	D17	D18	D19	
<i>gini</i>	2620	3201	3792	4503	5012	6500	7324	8342	9112	
<i>ks</i>	152	176	202	230	260	292	326	362	400	

We observed for all datasets that neither *ks* nor *gini* dominate the accuracy, measured on the test sample, of induced trees. It is increasing according to the numbers of categories. It varies between 60% and 95%. On time execution there is no dataset which could give a shorter time for *gini*, the execution time is relatively close for a few categories and deviates more and more when there are several categories.

We can say that from the significantly quicker time the kinds of trees that are built are quite different. So it is preferable to have an algorithm which gives the result more quickly. We have selected D9 (8-categories) to generate the induction trees for two metrics. The confusion matrix for the learning sample and the test sample obtained by both methods are shown in Tables 4 and 5.

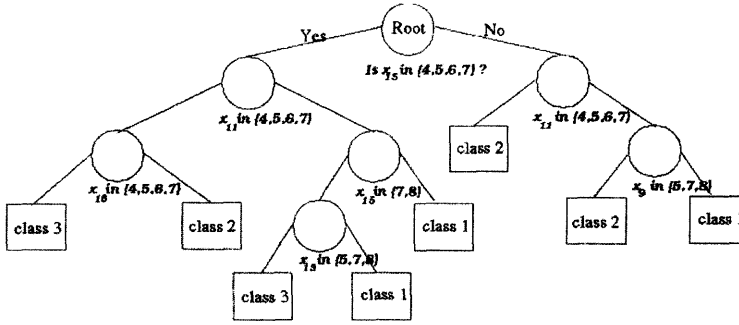
Table 4: Confusion matrix via *gini*

Learning sample						Test sample				
class assigned	True class					True class				
	class 1	class 2	class 3	Tot.		class 1	class 2	class 3	Tot.	
	class 1	65	4	6	75	class 1	866	122	179	1167
	class 2	19	91	11	121	class 2	384	1333	274	1991
	class 3	17	4	84	105	class 3	327	89	1127	1543
	Tot.	101	99	101	301	Tot.	1577	1544	1578	4699

Table 5: Confusion matrix via *ks*

Learning sample						Test sample				
class assigned	True class					True class				
	class 1	class 2	class 3	Tot.		class 1	class 2	class 3	Tot.	
	class 1	81	5	7	93	class 1	1070	197	191	1458
	class 2	11	68	9	88	class 2	238	1099	137	1474
	class 3	16	14	90	120	class 3	262	260	1245	1767
	Tot.	108	87	106	301	Tot.	1570	1556	1573	4699

The resulting tree obtained via *gini* criterion is shown in Figure 5 with a misclassification rate of 20.27% in the learning sample and 29.26% in the test sample.

Figure 5: Binary tree via *cart*

The resulting tree obtained using our criterion is shown in Figure 6 with a misclassification rate of 20.64% in the learning sample and 27.36% in the test sample.

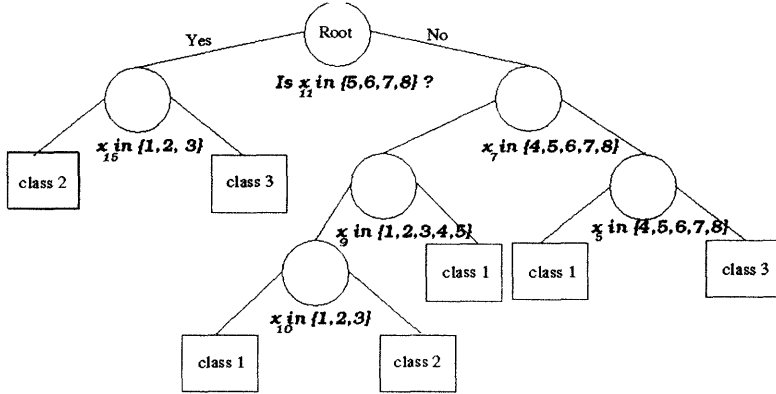


Figure 6: Binary tree via our method

There is approximately the same misclassification rate for both methods but the time to construct the tree by *cart* is exponential on a number of categories around $2^{8-1}(3-1) = 256$. With our method the corresponding run time is $(3-1)8(8-1)=112$. So there is a great gain in time for the same result.

We remark that the cutting threshold for some nodes in the tree via *cart* doesn't respects the natural order (discretisation of the quantitative variables) like the category $x_9 \in \{5,7,8\}$ and we do not know why this variable jumps the category encoded by 6. In our tree we do not find this situation.

4. Conclusion

The main aim of this study was to formulate new help for building segmentation trees and an efficient method regarding the time needed to find an optimal partition for constructing a binary tree. This strategy is proved only for our criterion [1] but we believe that it should be extended to other criteria.

References

- [1] Asseraf M. (1998), Extended Kolmogorov-Smirnov segmentation distance and optimisation for tree induction. Ph.D. dissertation, Université Dauphine, Paris, Fr.
- [2] Brandman Y. (1987), Spectral lower-bound techniques for logic circuits, Comput. Syst. Lab., Stanford, CA, Tech. Rep. CSL-TR-87-325.
- [3] Breiman L., Friedman J.H., Olshen R.A., and Stone C.J. (1984), Classification and regression trees, The Wadsworth Statistics/Probability Series, Belmont, CA.
- [4] Chou P.A. (1991), Optimal partitioning for classification and regression trees, IEEE Transaction on Pattern Analysis and Machine Intelligence, 13, 4, pp. 340-354.
- [5] Fayyad U.M. (1994), Branching on attribute values in decision tree generation, 12th National Conference on Artificial Intelligence (AAAI-94), pp. 601-606.
- [6] Fisher W.D. (1958), On grouping for maximum homogeneity, J. Amer. Stat. Assoc., 53, 789-798.
- [7] Friedman J.H. and Bentley J.L. and Finkel R.A. (1975), An algorithm for finding best matches in logarithmic time, Stanford linear Accelerator Center Rep. SLAC- UB-1549.
- [8] Henrichon J.E.G. and Fu K.S. (1969), A nonparametric partitioning procedure for pattern classification, IEEE Trans. Comput., C-18, 604-624.
- [9] Payne R.W. and Preece D.A. (1980), Identification keys and diagnostic tables : A review, J. Roy. Stat. Soc. A., 143, 253-292.
- [10] Quinlan J.R. Rivest R.L. (1989), Inferring decision trees using the minimum description length principal, Inform. Computat., 80, 227-248.
- [11] Quinlan J.R. Rivest R.L. (1993), C4.5: Programs for machine learning, Chapter 7, Grouping Attribute Values, Morgan Kaufmann.
- [12] Rounds E.M. (1980), A combined nonparametric approach to feature selection and binary decision tree design, Pattern Recognition, vol. 12, pp. 313-317.
- [13] Sethi I.K. and Sarvarayudu G.P.R. (1982), Hierarchical classifier design using mutual information, IEEE Trans. Pattern Anal. Machine Intell., PAMI-4, 441-445.
- [14] Stoller D.C. (1954), Univariate two-population distribution-free discrimination, J. Amer. Stat. Assoc., 49, 770-775.

Condition Monitoring, Root Cause Analysis and Decision Support on Urgency of Actions

G.Weidl

galia.weidl@secrc.abb.se

ABB Corporate Research & Mälardalen University, S-721 78 Västerås, Sweden

A.Madsen

Anders.L.Madsen@hugin.com

Hugin Expert A/S, Niels Jernes Vej 10, 9220 Aalborg, Denmark

E.Dahlquist

edt@mdh.se

ABB Process Industries & Mälardalen University, S-721 78 Västerås, Sweden

Abstract. We discuss the use of a hybrid system utilizing Object Oriented Bayesian networks and influence diagrams for probabilistic reasoning under uncertainties in industrial process operations. The Bayesian networks are used for condition monitoring and root cause analysis of process operation. The recommended decision sequence of corrective actions and observations is obtained following the “myopic” approach. The BN inference on most probable root cause is used in an influence diagram for taking decisions on urgency of corrective actions vs. delivery deadline. The build-in chain of causality from root cause to process faults can provide the user with explanation facility and a simulation tool of the effect of intended actions.

1. Introduction

There have been several attempts to develop process-monitoring systems for decision support on industrial process operation and control [1]. Process operators prefer a system, which can explain its inference conclusions and will help them to understand the underlying mechanism, which has caused a process abnormality. So, based on observations and automatically gathered evidence, the operators can steer the inference process, which quickly shows them the range of alternative decisions and their probable impact on the process efficiency. This decision support should use both technical input (e.g. failure prognoses) and financial information, such as expected utility and cost. Several teams are working in an industrial plant: maintenance crew, operators, process engineers, and management. Each team has its own responsibilities, objectives and its own optimization criteria. Therefore, a decision strategy requires a multi-criteria analysis.

The goal of our work is Root Cause Analysis (RCA) and Decision Support (DS). To reach the above objectives, we have chosen Bayesian Networks (BN) and Influence Diagrams (Id), since they incorporate probabilistic reasoning [2], [3], [4] and can provide explanations, which is especially suitable for decision support. Sequential adaptivity for such intelligent systems was developed in [5]. Decision-theoretic troubleshooting (DTTS)

for maintenance was discussed in [7]. A probabilistic approach to fault diagnostics in process operation and safety was suggested in [8], [9], [10]. The use of Object Oriented Networks (OONs notation for both OO BN and OO Id) for facilitation of the model construction for large and complex domains, and simpler modification of small-standardized BN fragments was proposed in [11]. Learning in OON was discussed in [12].

The contribution of this article is as follows. We model uncertainty in complex industrial processes by using knowledge-based probabilistic Bayesian networks and Influence diagrams. We construct OONs for repetitive structures in the plant. These OONs are used as fragments in the construction of OONs for RCA & DS on abnormality and failures in equipment and in process operation conditions. The last can be used as OON instances (subnets) within a larger network, in order to monitor the plant conditions and overall performance. For adaptivity to process changes we propose a kind of “supervised” sequential learning in BN. The RCA methodology is combined with the DTTS-algorithm to recommend an efficient sequence of actions and observations, and to reduce or prevent potential production losses. In addition, we extend the RCA methodology further to include decision support on urgency of corrective actions vs. delivery deadline. A similar utility-based approach can be used for selection of competitive decision alternatives for the same problem.

2. Problem formulation for RCA & DS

When process performance is in degradation, the chain of causes and events (see column 1 in Fig. 1) can represent the basic mechanism of a failure build up. It consists of root cause activation, which causes abnormal changes in the process conditions. Sensors or on-line calculations serving as soft sensors register abnormalities. This provides information to carry corrective actions by control or maintenance. If not corrected, these abnormal conditions can enable undesired events and cause a failure or other effects.

The aim of RCA is to find the root cause of a problem, while observing some early evidence. The next step of decision support should propose an (efficient) treatment as a sequence of corrective actions and observations. Treating just the effects or symptoms might be very inefficient and costly approach. To be able to accept or reject an advice on suitable corrective actions, the operator needs also some system explanations (in natural language or by picture) as motivation of the found root cause and proposed actions.

3. Hybrid Intelligent System. Overview

The decision analysis system comprises three main steps as shown in Fig. 1

- Root Cause Analysis and early warnings on abnormality, based on Bayesian networks. The BNs have conceptual layers, which incorporate the chain of causes and events.
- Decision Support on efficient sequence of actions, which is based on the DTTS algorithm for estimation of expected efficiency of actions by probability-cost function.
- Decision Support on urgency of alternative actions vs. order delivery deadline. It is based on Influence diagram for analysis of competitive decision options.

After gathering of evidence and signals preprocessing, the RCA & DS method include:

- Classification of signals into discrete states that is adaptive to process changes;
- Early risk assessment of abnormality based on the level-trend signal's pattern;
- RCA & DS by Bayesian Inference;

Exchange of information between RCA, DS, control, maintenance and production scheduling systems. The purpose is maximizing utility from production orders in the plant.

The model structure is incorporating, by construction, causal interpretation of the dependency relation between process variables, assets and other entities. This is used for explanation of inference conclusions to the user. Moreover, the built in causality of these

models provides the user with a simulation tool of the impact of intended corrective actions. The initially propagated information comprises signals of predictive character. These signals are gathered automatically from the Control System. In addition, for DS, information is gathered from the maintenance and production scheduling systems.

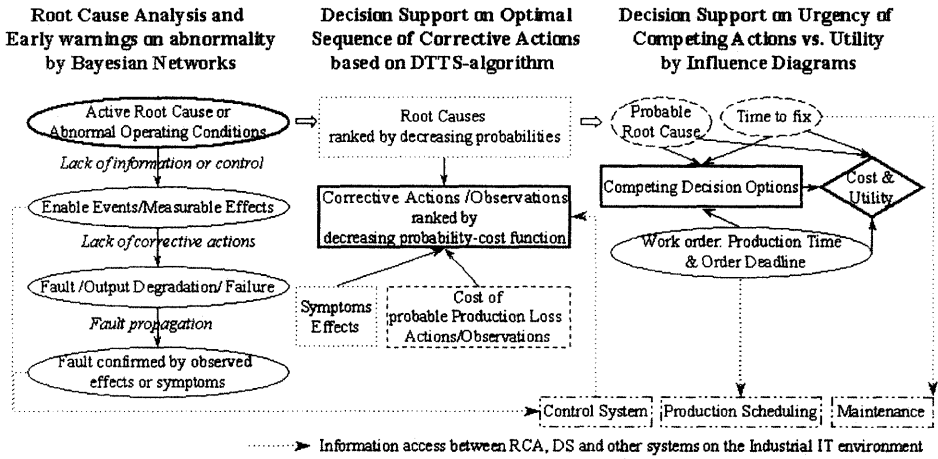


Fig. 1. The hybrid system for Root Cause Analysis, Decision Support on efficient sequence of Actions and Observations, and DS on Urgency of Competing Actions for the same root cause

Since the process behavior is changing over time, the quantitative (probability) learning in the models is based on sequential adaptivity [5]. The sequential adaptivity is combined with supervised learning, where process effects are gathered automatically from the Control System and are used to confirm or reject inference conclusions. Thus, the adaptation in industrial environment is reflecting the actual variables' configuration for the observed fault. Moreover, the adaptivity algorithm is supervised by the user feedback supplied as evidence on the actually found root cause of a particular set of process parameters. This is performed by combining the initial set of evidence of predictive character with findings on the actual root cause and with findings on the confirming events.

4. Bayesian Networks and Influence diagrams for RCA & DS

A Bayesian Network (BN) consists of a directed acyclic graph (DAG) over chance nodes. The chance nodes represent random variables. The strength of dependency relations between the chance nodes is quantified by conditional probability distributions (CPD).

An Influence diagram (Id) is an extension of a BN. An Id consists of a DAG over chance nodes, decision nodes and utility nodes with the following structural properties [3]:

a) a directed path should comprise all decision nodes; b) the utility nodes have no children.

For the quantitative specification it is required that

- the decision nodes and the chance nodes have a finite set of mutually exclusive and exhaustive states; the utility nodes have no states;
- to each chance node A is attached a conditional probability table $P(A/pa(A))$, where $pa(A)$ are the parents of A ;
- to each utility node V is attached real-valued function over $pa(V)$.

In the models, we use the following notations: chance nodes are indicated by ellipses, decision nodes - by squares, and utility nodes by diamonds, dependency relations between nodes are indicated by directed links, see Fig. 2. A link into a chance node represents

probabilistic dependence, whereas a link into decision node specifies that the parent is observed prior to making the decision on the child.

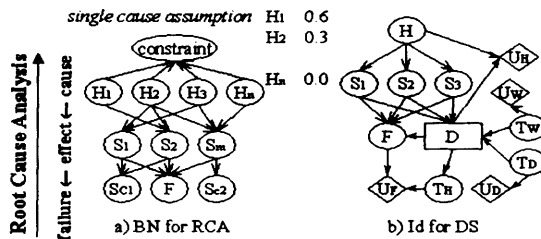


Fig. 2. Schematic structures of a) Bayesian Network for RCA; b) Influence Diagram for DS

For RCA and DS, the initial qualitative structure of the network (BN or Id) is constructed from process knowledge. To this initial structure are added more dependency relations extracted by analysis of data from normal and abnormal process operation.

To provide the user with explanations and decision support, the inference computation are performed on discretized variables, which states are naturally labeled by "words", e.g. high, normal, low, increased, steady, decreased, asset broken, equipment malfunctioning, etc., where the strength of a statement is quantified by its probability. For construction of the BN & Id models, inference and probability update we have used the Hugin tool [6].

In the BN models for RCA, H_i represent the hypotheses on root causes, S_i - their effects and symptoms, F - the consequent failures, S_c are confirming events or deviations from desired prerequisites. All root cause nodes are constraint by the single fault assumption. This is a reasonable assumption in industrial operation, since multiple faults can be treated one at a time. Alternatively, all possible root causes are represented as states of one H node. Usually, the findings are on F , S_c and S_i . With this information, the RCA is reasoning about the state of H .

In Fig. 2b), the decision node D represent an action controlled outside the influence diagram, i.e. it is not given a probability distribution. The chance nodes T_i add time constraints on the decision options. The utility nodes U_i represent local utility functions of the Id. All possible corrective actions are represented as states of one decision node D . We assume that only one corrective action can solve a problem due to a certain root cause. This is expressed by the utility (cost)-function U_H for the right combination of a root cause and its corresponding corrective action. In case there are other utilities in the influence diagram, this may produce a problem. The usual approach (if there are other utilities) is to associate a high negative value with the combinations of corrective actions and actions which are undesired.

The root causes of abnormality may originate from various levels in the plant hierarchy:

- Basic Assets (e.g. sensors, actuators (valves, pumps), screens inside digester)
- Equipment Units (e.g. heat exchangers, evaporators, pulp screens, etc.)
- Maintenance Conditions
- Process Operation Conditions
- Plant Conditions (How the equipment is used in the whole process concept?)

Unsuitable process conditions, e.g. overload working conditions, might cause malfunction of assets or equipment. Early detection of abnormal exploitation conditions is important for predictive maintenance and control. Thus, the system should provide early warnings of abnormality and all deviations from normal behavior can be corrected at initial stage.

There exist a number of data-driven packages for diagnostics of basic assets. These packages should be exploited. Therefore, the BN is using a hybrid cross-combination of their outputs as evidence on the uncertainty or reliability of input information for RCA [10]. The

value added by Root Cause Analysis will be at process and plant level, especially since it can provide reusable design of process sections within the same process industry. The reusable design can be ensured by the use of object oriented BNs.

5. Object-Oriented Networks (OON) for modeling of complex systems

During decision-making and RCA of faulty operation in industrial complex systems, there is a repeated change of reasoning between basic assets, equipment, and maintenance and process operation conditions. The decision difficulties are set by the human ability to capture all details of a complex system simultaneously. This requires support in BN models development, while working on different levels of the plant hierarchy abstraction.

Knowledge acquisition (and learning) for BN can become a time consuming task for large industrial systems, if not performed efficiently. We focus here on methods and tools supporting the model development, BN-fragment modifications and inference for root cause analysis and multi-criteria decision analysis. We use OON¹ for top-down RCA of industrial systems, which allows different levels of modeling abstraction reflecting different levels of plant hierarchy. Describing a network reflecting the plant hierarchy, provides a clear model overview and thus gives a better basis for knowledge acquisition and exchange of ideas between engineers, operators and maintenance crew.

5.1. Object-Oriented Network for repetitive structures in RCA

Industrial complex systems are often composed of collections of identical or almost identical components (e.g. control loops and basic process assets, like sensors, valves and pumps). Therefore, models of industrial systems often contain repetitive patterns (i.e., common solutions of typical problems). In Bayesian networks and influence diagrams, such patterns are network fragments. The use of instance nodes facilitates significantly the construction of multiple instances of a network fragment [11]. We use this idea to model (measured and computed) signals uncertainties, signals level-trend classifications and control loops, as small standardized sub-OON or fragments within the OON of the problem domain.

Fig. 3 shows the *OON for control loops*. It can be used to represent the control loops of general interest to process industry, e.g. pressure, flow, level, temperature control. The sensor reading from any DCS-measurement is conditionally dependent on random changes in two variables: real value under measurement and sensor status of the instrument. Its probability distribution is a mixture of normal and uniform distributions for the real value when the sensor status is true or false respectively [10]. We use mixture of Gaussian distributions as CPD on selected soft interval states to represent the most characteristic values of a continuous signal during normal and faulty operation. Failures are events of low

¹ An Object-Oriented Network is a network (i.e., Bayesian network or influence diagram) that, in addition to the usual nodes, contains instance nodes. An instance node is a node representing an instance of another network (i.e. subnet). Of course, the network of which instances exist in other networks can itself contain instance nodes, whereby an object-oriented network can be viewed as a hierarchical description (or model) of a problem domain. An instance node connects to other nodes via *interface nodes*. In order to provide clear overview, the instance nodes are hiding detailed information (on dependency between the variables) inside its structure. Therefore, the interface nodes usually comprise a subset of the nodes in the instance. Interface nodes are subdivided into *input (child) nodes* and *output (parent) nodes*, see Fig. 3. Input nodes are placeholders for (basic) nodes outside the instance. Output nodes gives access to nodes inside the instance. An output node can be specified as parents of nodes in the network containing the instance node or can be bound to an input node of another instance node of the network. In OON, we use the following *notations*: instance nodes are squares with input and output interfaces: input nodes are ellipses with shadow dashed line borders, output nodes are ellipses with shadow bold line borders, see Fig. 3.

frequency. Therefore, mixture of Gaussian distribution during normal behaviour and Poisson distribution during faulty operation is suitable to express the CPD of "real_value".

A malfunctioning actuator (i.e. valve or pump) is a root cause from the group of basic assets. The *OON* for basic process assets can be obtained from the model in Fig. 3 by simple extension incorporating root causes due to equipment and other basic components (e.g. pumps and screens). Adding extra states in the node "status_Actuator" in order to represent other possible root causes incorporates this. The findings on the status of sensors, actuators and the operation mode settings are provided by other diagnostic packages.

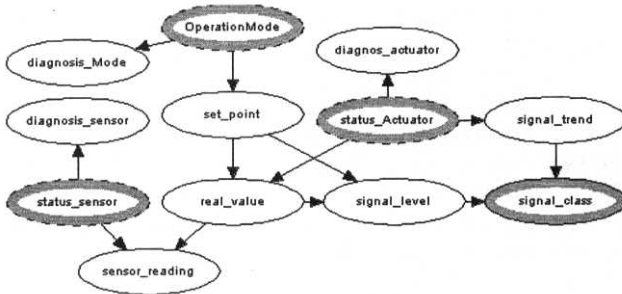


Fig. 3. OON model of control loops and signal classification, based on level-trend pattern.

The *OON* for risk assessment of process abnormality is given in Fig. 4. It will indicate improper operation conditions, recognized in changed level-trend pattern. A combination of several such *OON* models allows us to perform RCA of abnormalities observed in process conditions, and to confirm effects of failure events, which are used for sequential adaptive learning from data [5]. By using supervised sequential learning in *OON*, each instance node performs its own adaptation, which is preferable in order to take into account individual conditional probability distributions for subnets of the same *OON* structure, but placed in different context or position in the industrial process.

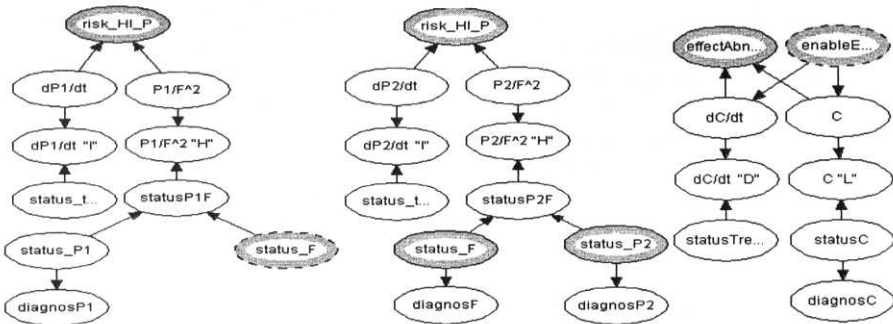


Fig. 4. OON models: a) - b) for risk assessment of abnormal process conditions, which can enable undesired events; c) for assessment of abnormality in effects from events. The effects are used as feedback in the learning algorithm to confirm a possible root cause of abnormality. Note that Fig. 4b) is an extension of Fig. 4a) with a single node (diagnos_F). This exploits the property of inheritance in OON. Inheritance simplifies the construction of OON, see Fig. 5.

In Fig. 4, there can be findings on all nodes in the network, except for the purely modeling nodes (statusP1F, statusP2F). The modeling nodes are used to simplify the construction of the network.

Fig. 5 - Fig. 6 show how we use the above sub-OONs as building BN-fragments in order to represent the entire problem domain at different levels of RCA abstraction. The OON models incorporate the consequent causality steps of the basic mechanism of a failure build up, as described in section 2. Fig. 5 shows an OOBN model of an event (e.g. pump plug), which is enabled by abnormal process conditions (e.g. high flow concentration) and is confirmed by another event (e.g. low pump capacity). Fig. 6 shows an OON modeling several events (e.g. various pump problems), which can cause process faults or failures (e.g. if a pump fails and it is indispensable for process operation, the plant should be shut down).

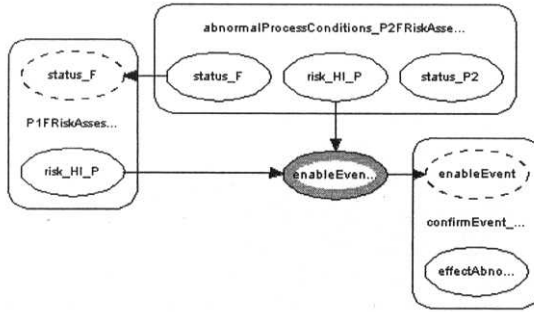


Fig. 5. OON model representing a configuration of different abnormality conditions in process operation, which can enable undesired events in basic assets, equipment or process operation. The effects of such events confirm or reject inference conclusions and serve as learning feedback.

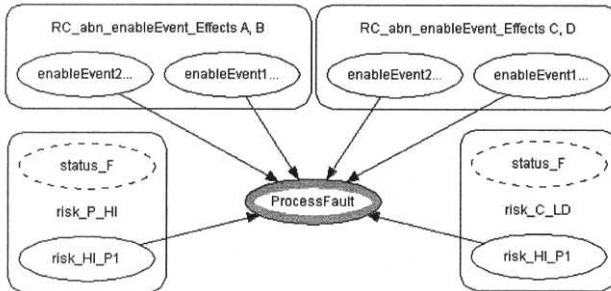


Fig. 6. OON model representing a configuration of different undesired events, which can cause a process fault, failure or undesired deviation in process output

6. RCA & DS on Efficient Sequence of Actions and Observations

The purpose of RCA is to find the origin of abnormality at an early stage and provide decision support on suitable corrective actions for control and maintenance. If the system goal was to provide only an advice on suitable corrective actions for each root cause, the influence diagram of Fig. 2 can provide a solution. Then the sequence of corrective actions will follow the ranking of root causes after decreased probability/cost. But some of the symptoms might not be measured and will require observations. The order of decisions might also be important for retaining of optimal production rate and reducing the cost of maintenance or the cost from potential unplanned production stops. Therefore, ranking is performed on the base of the *probability-cost* (efficiency) *function*, which is computed for

the corrective actions and not for the root causes. The ranking is used to find the optimal sequence of corrective actions (and observations) on a probable root cause. This will ensure the minimal cost of maintenance or minimal cost due to potential production losses.

The origin of process disturbance can be due to abnormality in process conditions or maintenance. Usually, when the industrial process performance is in degradation, we have

- a set of evidence E , including early warnings, symptoms and user observations,
- single or multiple faults (root causes), (we will assume single fault),
- a number of corrective actions $D_i = \{\cup D_i^{(control)}\} \cup \{\cup D_i^{(maintenance)}\}$,
- each corrective action has a cost $C_i = \text{Cost}(D_i, E)$. We will assume $C_i = \text{Cost}(D_i)$, i.e. the cost is independent of the evidence.

The task is to determine the optimal sequence of corrective actions $S = \langle D_1, D_2, \dots, D_N \rangle$ to solve the problem at as low cost (or as low production loss) as possible. We follow the “greedy” approach to optimal sequence of actions [7]. Notice that the sequence is only optimal under the single fault assumption.

In general, the expected cost of a sequence of corrective actions is given as

$$EC(S, E) = \sum_{i=1..n} P_i(E_{i-1}) C_j(D_i, E_{i-1}), \quad (1)$$

where $P_i(E_{i-1})$ is the probability that the previous actions did not solve the problem.

Under the single fault assumption and the assumption of cost function independence on the order of actions, the expected cost (1) becomes: $EC(S, E) = \sum_{i=1..n} (1 - \sum_{j < i} P_j) C_j(D_i)$, where P_i is the probability that action D_i solves the problem given $S' = \langle D_1, D_2, \dots, D_{i-1} \rangle$ did not, since we assume perfect actions. Then, the recommended decision sequence will incorporate actions, which are sorted in decreasing order of efficiency

- $P_i / C_i \geq P_{i+1} / C_{i+1}$, for optimal sequence of *repair* actions, when the failure is a fact,
- $P_i C_i \geq P_{i+1} C_{i+1}$, for sequence of *preventive* actions, reducing the potential production losses.

The decision algorithm then follows the myopic approach to calculating the expected value of observation O

$$EV(O) = EC(S', E) - \sum_{i=1..n} P(O=o_i | E) EC(S', E \cup \{O=o_i\})$$

where $EC(S', E)$ is the expected cost of maintenance (or production loss) if O is not observed. In general an observation on O is made, if $EV(O) > C(O)$, otherwise the action with highest efficiency is performed and the set E is updated. We choose to make the observation with the highest expected benefit $EV(O) - C(O)$. This algorithm proceeds in a loop until the problem is solved. As shown in Fig. 1, the user is advised on the efficient sequence of suitable actions, together with the RCA estimations on root causes.

7. Influence diagrams for deciding on the urgency of actions

An efficient system should also provide the operator with decision support on the urgency of proposed corrective actions for a particular root cause, especially when competing decision options might have different technical or economical impact.

7.1. Urgency of pump maintenance vs. order delivery deadline

During process operation, an operator is provided with an early warning from the RCA system, pointing at a pump problem (e.g. plugging). Dependent on process equipment condition, it might take different time to fix the problem. If not fixed on time, it might require process stop. On the other hand, the production is planned for just in time delivery and the work order is approaching its deadline. Taking the plant out of operation might delay the

delivery. In order to have a clear risk assessment and make the right decision, the following information is taken into account:

- What is the most probable root cause of abnormality/fault?
- What is the estimated time, left for production fulfilling the work order?
- When is the work order deadline?
- How long is the estimated time needed to fix the failure?
- Would the delivery be delayed and with how many delay hours?
- What is the cost of delays?
- What is the expected payment for the delivery, under the condition of taking different maintenance decisions?

The main decision strategy is acting to maximize the company benefit. The inference results provide decision support on: "Which corrective action to take: repair, replace or wait?"

After the order deadline is reached, the Id parameters are updated based on user feedback: "What was the actually used time for problem fixing?"

The above outlines a scenario of decision support on the urgency of corrective actions. It is modeled as an influence diagram, see Fig. 7. It shows one possible scenario of working system with the evidence entered and the expected utilities of each decision option computed. The probability of pump break is dependent on the chosen decision.

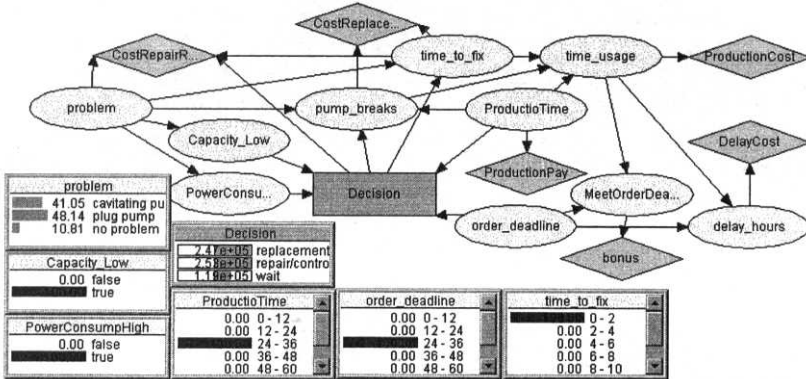


Fig. 7. Id for DS on urgency of maintenance action vs. delivery deadline. The monitors illustrate a scenario of a working system with nodes instantiated by evidence (from other systems): capacity & power consumption (control system evidence), time to fix the problem = 0-2 h (maintenance or RCA finding), production time = order deadline = 24-36 h (production scheduling). The user can also enter the evidence.

All continuous variables are discretized on numerical intervals. The conditional probability of the needed time to fix the problem is dependent on the alternative decision options. It is expressed as uniform CPD on the discretized time interval from t_{min} to t_{max} ($t_{min} < t_{max}$) needed for a certain action

$$P(\text{time_to_fix} / \text{problem}, \text{Decision}) = \begin{cases} \text{Uniform}(\min, \max), & \text{if } \text{Decision} = \{\text{repair}, \text{replacement}, \text{wait}\} \\ & \& \text{problem} = \{\text{"capacity"}, \text{"power_consumption"}, \text{"no problem"}\} \\ 0, & \text{if } \text{Decision} = \text{"wait"} \end{cases}$$

The decision "wait" implies that no actions are undertaken before order deadline is reached.

The optimal decision is determined based on several utility functions, which are given as

$$\text{Utility}(D=\text{repair}/\text{problem}, \text{Decision}) = (-\text{time_tax} - \text{Cost}_{\text{lost_production/unit time}}) * \text{time_to_fix}$$

$Utility(D=replace|problem, Decision) = (-time_tax - Cost_{lost_production/unit\ time}) * time_to_fix - Cost_{pump}$
 $Utility(production\ cost) = (Production\ cost\ per\ unit\ time = -20.000\$) * time_usage$
 $Utility(delay\ cost) = delay_hours * (delay\ cost = -(10\%)*order_volume)$
 $Utility(meet\ order\ deadline) = Bonus\ for\ delivery\ on\ time = 50.000\$$
 $Utility(production\ payment\ from\ customer) = production\ time\ tax * production_time$

This case is not only pump specific. It can be used in general for deciding on the urgency of maintenance actions for any basic process asset or equipment. The difference would be in the specific root causes (problems) and on their corresponding cost and utility functions on the base of which the decision is taken.

Conclusions

We have discussed the construction and interactions between Bayesian networks (for RCA) and influence diagrams (for DS). We propose an influence diagram model, in order to combine the technological objectives of the industry (e.g. process maintenance, reliable operation and safety) with the financial criteria for the economy (maximizing company benefit and customer satisfaction). The decision support example can be generalized to utility-based selection of competitive decision alternatives to the same problem.

"Supervised" sequential learning takes into account feedback (from user and control system). It allows the RCA system to adapt to process changes and to reflect individual equipment behaviour. It accounts for individual conditional probability distributions in subsets of the same OON structure, but at different positions in the industrial process.

The outlined methodology is under deployment on pulp & paper process applications and it could be generally applicable in process industries. The models development and modifications are facilitated by interconnected object oriented networks. This will provide RCA information between different levels in the plant hierarchy and can be used for monitoring of process conditions and plant performance. The proposed system fits naturally in the Industrial IT environment of the plant, which is an object-oriented platform.

References

- [1] Olsson G. (1995): "Development of the operators current work with control and surveillance of processes and industrial plants", NUTEK's Analysis report, Uppsala, Sweden
- [2] Pearl J. (1988): Probabilistic Reasoning in Intelligent Systems, Morgan Kaufmann Publishers
- [3] Jensen F.V. (2001): Bayesian Networks and Decision Graphs, Springer Verlag, New York
- [4] Cowell R.G., Dawid A.P., Lauritzen S.L., Spiegelhalter D.G. (1999): Probabilistic Networks and Expert Systems, Springer Verlag
- [5] Olesen K.G., Lauritzen S.L., Jensen F.V. (1992): aHugin: System creating adaptive causal probabilistic networks, UAI 92: 223-229
- [6] <http://www.hugin.com>
- [7] Heckerman D., Breese J., Rommelse K (1995): "Decision-theoretic Troubleshooting" Communications of the ACM, 38(3): 49-56
- [8] Leung D., Romagnoli J. (2000): "Dynamic probabilistic model-based expert system for fault diagnosis", Computers and Chemical Engineering 24: 2473-2492
- [9] Carlsson T. (2002) Intelligent Systems in Process Safety, Licentiat Thesis, Royal Institute of Technology, Stockholm, Sweden, ISSN 1402-7615
- [10] Weidl G., Dahlquist E. (2002) Root Cause Analysis for Pulp and Paper Applications, In proceedings of 10th Control Systems conference, pp.343-347, Stockholm; Sweden, June 3-5, 2002
- [11] Koller D. and Pfeffer A (1997), Object-oriented Bayesian networks, In Proc. UAI-97, pp. 302-313, Morgan Kaufmann
- [12] Langseth H. and O. Bangsø (2001) Parameter Learning in Object Oriented Bayesian Networks, In Annals of Mathematics and Artificial Intelligence

Towards a Unique Framework to Describe and Compare Diagnosis Approaches

Cecilia H. ZANNI, Marc LE GOC, Claudia S. FRYDMAN
LSIS - UMR - CNRS 6168 - Université d'Aix-Marseille III
13397 Marseille Cedex 20 - France

Abstract. This paper introduces a unique framework to describe the different approaches to intelligent monitoring and diagnosis present in the literature. In first place, the state of the art in diagnosis is reviewed and then we propose a framework for analyzing the approaches presented so far, based on the KADS standard for development of knowledge based systems. In the end, we present our conclusions on the conceptual level of description of these systems, which lead us to state a general structure for them.

1. Introduction

The literature is almost unanimous in recognizing the existence of a major problem in the specification and design of large and complex reactive systems [6]. A reactive system is characterized by being, to a large extent, event-driven, continuously having to react to external and internal stimuli. Examples include cars, communication networks, computer operating systems, the man-machine interface of many kinds of software, and many other complex industrial processes, such as the operation of blast furnaces, for example [10].

Since there is no operational mathematical model for the dynamics of reactive complex continuous processes, the usual means of control theories cannot be used to design an automatic controller, suited for such complex systems. In addition, processes required to reach a very high level of economic performance tend to be complex.

In the past three decades, however, several approaches have been proposed for dealing with this complexity. One consists of compensating the lack of a mathematical model for the process dynamics by the knowledge used by the human operator who controls those complex processes.

Recent advances in different computer sciences, such as artificial intelligence and simulation, propose methods, techniques and tools to exploit such informal knowledge, and new kind of controllers have been developed since 1980 [8].

In this paper, we aim at comparing different approaches for monitoring and diagnosis of continuous processes, whose behavioral models come from expert knowledge. We propose, then, to adopt a unique conceptual model for them.

In first place, we recall the general definition of diagnosis and we describe the state of the art in intelligent monitoring and diagnosis of industrial systems, namely the heuristic-based, the model-based and the task-based approaches.

In a second place, we propose a method for comparing the approaches reviewed so far, as the level of description of them vary and make difficult to understand the differences and

similarities among them. This framework is based on the KADS standard for knowledge based systems development.

In the end, we show our conclusions, facing to find a general description for these systems.

2. Diagnosis Problem Solving

Diagnosis was a fundamental area for Artificial Intelligence since the 70's. Many Artificial Intelligence methodologies originated from diagnosis developments and then spread to other areas of interest. On the other side, diagnosis has always been a blend of theoretical and experimental research [7].

The diagnosis problem can be formulated as follows:

Given:

- A system (device, physical system, physiological system, ...),
- A set of observations (measurements, tests, symptoms, examinations ...) corresponding to abnormal (unexpected, anomalous ...) behavior,

It is expected to determine what is wrong with the system in order to re-establish the system normal behavior (therapy, repair, reconfiguration, ...) [7].

Since the 70's, a large number of diagnosis systems have been described, and it is very difficult to compare them because of the multiple points of view, techniques, tools and methods adopted in the design of them.

3. State of the Art in Diagnosis

3.1 Heuristic Based Approach

The basic assumption of this approach (70's) is that diagnosis is a heuristic process. It means that experts rely on associational knowledge of the form symptoms → faults (diseases). This kind of knowledge derives from experience. So it must be elicited from domain experts and represented using suitable knowledge representation languages (Figure 1).

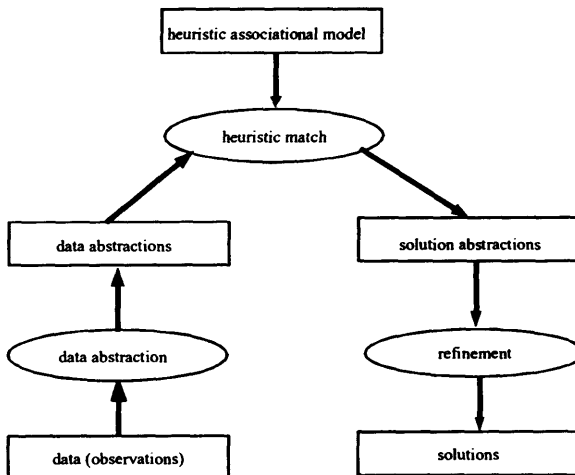


Figure 1. Diagnosis as a heuristic classification [7]

Examples in the literature: Mycin (Stanford Univ. 72/79) [30], Delta-Cats1 (General Electric) [31], PIP (MIT, 72-78). [7]

3.2 Model Based Approach

The basic paradigm of the model-based approach (late 70's - beginning of the 80's) can be understood as the interaction of observations and predictions (Figure 2) [5].

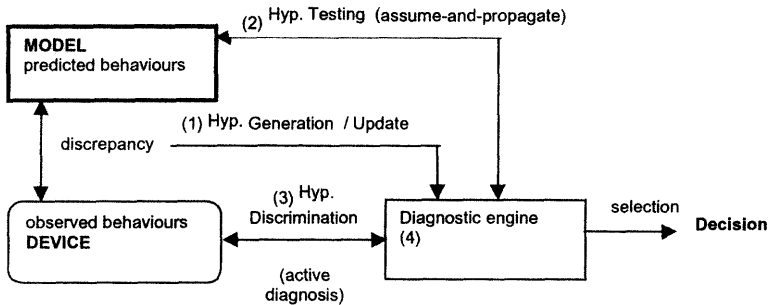


Figure 2. Classical architecture of a diagnosis system relying on a model

This approach covers:

- *Symbolic Models for Static Diagnosis*
 - CONSISTENCY-BASED DIAGNOSIS
Examples in the literature: HT of Davis, DART of Genesereth, GDE of de Kleer and Williams, GDE+ of Struss and Dressler, Sherlock of de Kleer and Williams [3].
 - ABDUCTIVE DIAGNOSIS
Examples in the literature: Cover & Differentiate of Eshelman, CHECK of Console and Torasso [3].
- *Symbolic Models for Dynamic Diagnosis*
 - Associative models
 - EXPERT SYSTEMS
Examples in the literature: IFP (Institut Français du Pétrole) with the Alexip software [13]; PICON and G2, an expert system generator [14]; France Telecom for monitoring the Transpac network using the Chronos software [2], RTWorks and CogSys [15]
 - PATTERN RECOGNITION
Examples in the literature: AUSTRAL project [29]; GASPARD project [2], IxTet in the framework of project Esprit Tiger [16].
 - Predictive models
 - QUALITATIVE MODELS
Examples in the literature: MIMIC, a monitoring system [2], CA-EN system in the frame of the Esprit Tiger project for the monitoring of gas turbines [17].
 - DISCRETE MODELS
Examples in the literature: ESSO refinery in Canada [18]; monitoring of the Transpac network [20, 21, 22, 28]
 - Explanatory models
 - INFLUENCE GRAPHS
Examples in the literature: Esprit Alliance project [23], CA-EN software in Tiger project [17].
 - CAUSAL GRAPHS
Examples in the literature: Matra Marconi Space for its satellite monitoring software [24, 25, 26], DIAPO system for diagnosing the cool and pump sets in EDF nuclear power plants [27].

3.3 Task-Oriented Approach

The task-based approach (beginning of the 90's) aims at modeling problem solving behavior in terms of the knowledge that is used for the problem solving. Thus, regardless how a diagnostic system is implemented (rule based, frame based, connection based), it is possible to focus on the goal of the system and on the knowledge that is applied to achieve the goal. This modeling is reached by identifying tasks at various levels of abstraction above the implementation level (Figure 3)

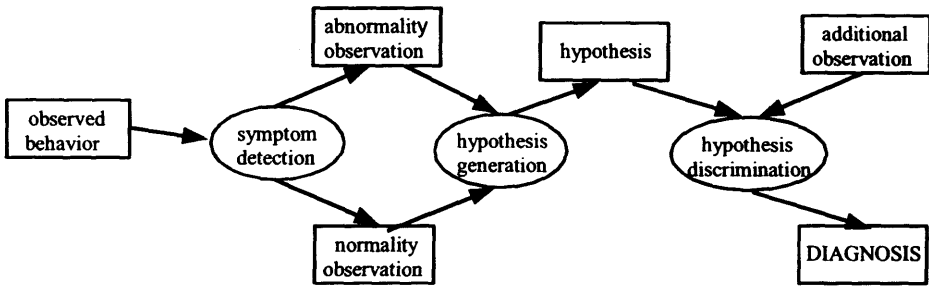


Figure 3. Task-oriented approach to diagnosis

Examples in the literature: Generic Tasks [32], CommonKADS [11,12], Problem Solving Methods [33], Components of Expertise [34; 35], Firefighter [36], Protégé-II [37], Mike [38], Task [39].

Within this framework, the placing of the Sachem system is difficult [8], [10]. Considering the absence of dynamic models of blast furnaces, Sachem might be seen as a kind of heuristic system (heuristic based approach). But Sachem has been designed with the KADS methodology according to a task-oriented approach. And on the other side, Sachem implements the same pattern recognition techniques as Tiger, to use associative models of blast furnaces behavior (model based approach). So, the Sachem example shows that the preceding categories are not enough to describe the complexity of actual diagnosis systems.

4. Our Framework for Analysis

Having reviewed the state of the art of intelligent diagnosis present in the literature, the main remark to note is that the levels of description of each approach vary (some of the approaches show aspects for modeling, but others face the resolution of the problem). This way it is hard to have a real comparison of the goals and scope of each of them.

Due to the fact that CommonKADS is a standard for development of knowledge based systems, it might be used for describing any knowledge-based system. Therefore, we have decided to describe the approaches in Section 3, according to it.

The CommonKADS abstraction cycle is based on a *conceptual model*, CM, and makes a distinction among this model and the Functional Model (FM) and the Design Model (DM) (see Figure 4).

This general framework makes a specific differentiation among the Conceptual Model, the Functional Model and the Design Model, each of them representing different views of the system to be conceived.

The main characteristics of each model are described as follows:

- The Conceptual Model is supposed to contain all the knowledge, and may be validated by the "client". It lies at the knowledge level.

- The Functional Model is supposed to contain all the functions, and may be validated by the development team. It lies at the symbol level.
- The Design Model includes all the techniques used to perform the functions described at the functional level. It also lies at the symbol level.

These three models gives us three levels of description that will allow us to perform a comparative analysis on the different approaches for diagnosis present in the literature.:

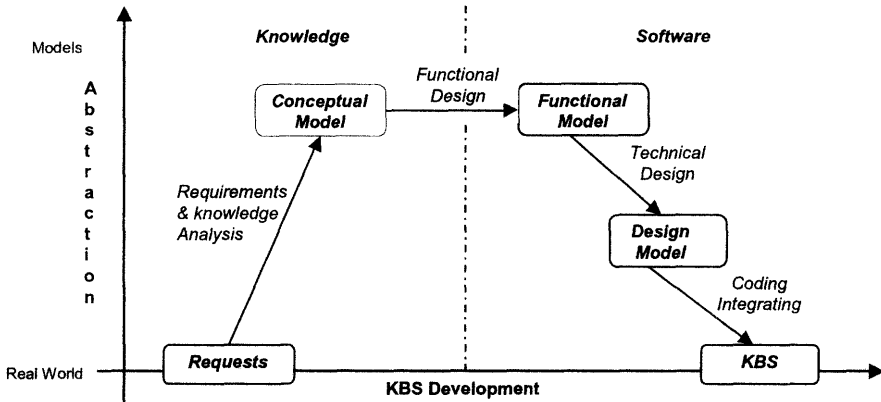


Figure 4. Abstraction cycle in CommonKADS

- *Conceptual Level* (associated with the Conceptual Model): At which we describe the nature of the knowledge source and the nature of the reasoning process.
- *Functional Level* (associated with the Functional Model): At which we describe the nature of the models or, in other way, how causality is represented.
- *Technical Level* (associated with the Design Model): At which we describe the nature of the underlying theory for representing and exploiting the knowledge (technical calculus or computation).

In this paper, only the conceptual level of description is considered. The next section proposes a unique conceptual framework that will allow placing the different diagnosis systems.

5. Towards a Unique Conceptual Framework

We propose to reconsider the state of the art in intelligent diagnosis approaches according to those 3 levels of description, explained in Section 4.

The description at the conceptual level first leads to elicit and to compare the basic inference structure adopted in the different diagnosis approaches. Next, a unifying inference structure is proposed, that permits to describe all the approaches as a specialization of this unifying inference structure.

Regarding the conceptual level, we have the following structures, each of them associated with one or more of the approaches revisited.

- *Heuristic System Model*
Depicted in Figure 5. Present in the *Heuristic Approach* and in the *Recognition Based Approach*.
- *"Predict & Compare"*
Depicted in Figure 6. Present in *Consistency-based diagnosis* and in *Discrete Models*.

- “Observe & Explain”
Depicted in Figure 7. Present in *Abductive diagnosis*.
- “Tracking & Interpretation”
Depicted in Figure 8. Present in *Qualitative Models*.
- “Symptom detection, Hypothesis generation, Hypothesis discrimination”
Depicted in Figure 9. Present in *Task-oriented approaches*.

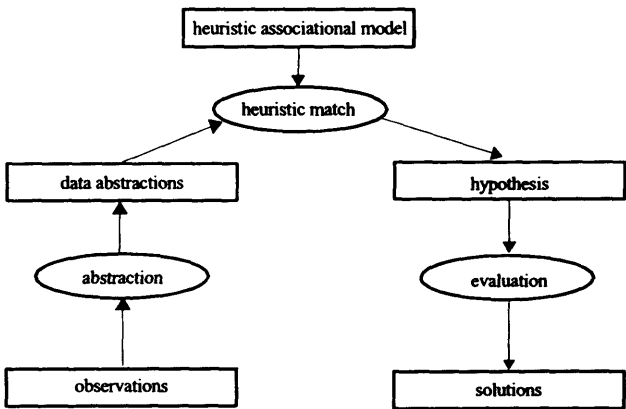


Figure 5. Heuristic System Inference Structure

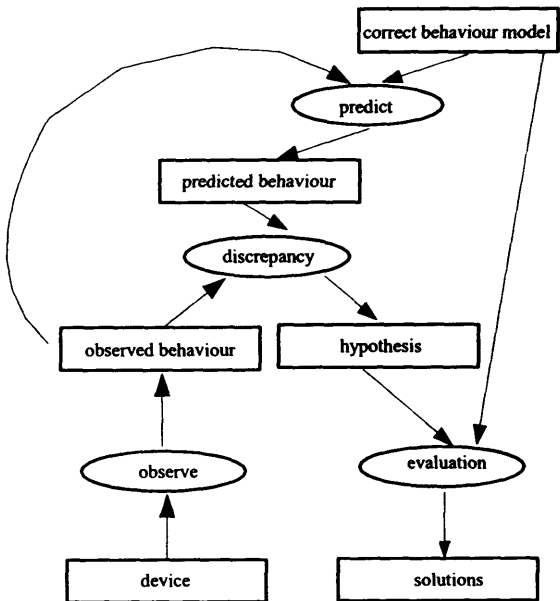


Figure 6. "Predict & Compare" Inference Structure

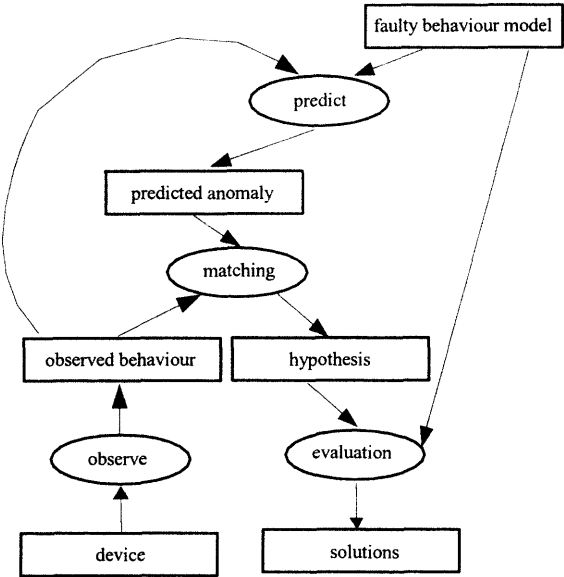


Figure 7. "Observe & Explain" Inference Structure

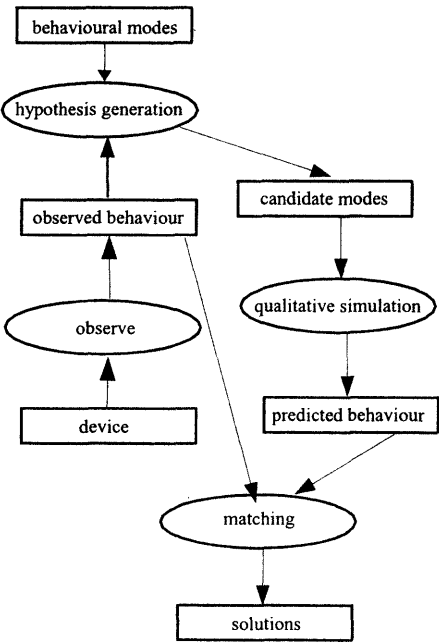


Figure 8. "Tracking & Interpretation" Inference Structure

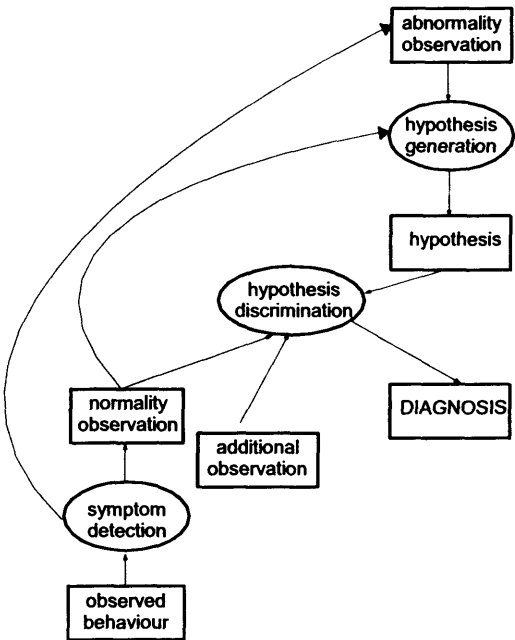


Figure 9. “Symptom detection, Hypothesis generation, Hypothesis discrimination” Inference Structure

Regarding the conceptual level of description, Figures 5 to 9 may be generalized as in Figure 10. This leads us to a unique description of the Intelligent Diagnosis task, regardless of the considered approach.

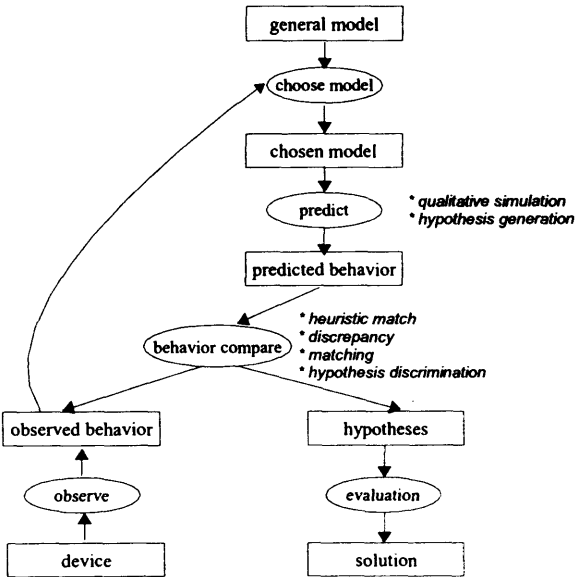


Figure 10. General description of Diagnosis at the Conceptual Level

Figure 10 results from the following facts:

- The “predict” inference in Figures 6, 7 and 10 is conceptually equivalent to the “qualitative simulation” inference in Figure 8 and to the “hypothesis generation” inference in Figure 9.
- The “behavior compare” inference in Figure 10 is conceptually equivalent to the “heuristic match” inference in Figure 5, to the “discrepancy” inference in Figure 6, to the “matching” inference in Figures 7 and 8 and, finally, to the “hypothesis discrimination” inference in Figure 9.
- Common concepts in Figures 5 to 10 are “Observed Behavior”, “Hypothesis”, “Predicted Behavior”, “Solution”. Figures 6 to 10 also have the concept “Device” in common.
- Each inference structure refers to a notion of “System Model” (“*Heuristic System*”, “*Tracking & Interpretation*”) to capture the knowledge of the correct behavior (“*Predict & Compare*”) or the faulty behavior (“*Observe & Explain*”). In the Task-Oriented approach, the model of the system is distributed between the “hypothesis generation” and the “symptom detection” inferences.

6. Conclusions

In this work, we recalled the general definition of the diagnosis concept, and reviewed the state of the art present in the literature.

A general framework for comparing the visited approaches has been proposed. The description at the conceptual level has been developed, showing the existence of a unifying Inference Structure. The advantage of this “Unifying Inference Structure” is that each visited Inference Structure can be seen as a specialization of the latter. It constitutes then a general description of the visited Diagnosis Systems at the conceptual level.

Our future work will deal with SACHEM systems [8], showing that it does not fit within this general description.

References

- [1] Vidal, Thierry - "Course on "Dynamic Model-Based Diagnosis" - <http://www.ida.liu.se/~thivi/Diag> - 1997
- [2] Cauvin, S. et al - "Monitoring and alarm interpretation in industrial environments" - AICOMS Vol. 11-3-4, p. 139-173 - 1998
- [3] Benjamins, Richard - "Problem Solving Methods for Diagnosis" - Ph. D. Thesis - University of Amsterdam - The Netherlands - June 1993
- [4] Benjamins, Richard - "On a Role of Problem Solving Methods in Knowledge Acquisition - Experiments with Diagnostic Strategies" - Laboratory of Integrated Systems - Escola Politécnica of the University of Sao Paulo, Brazil - Proceedings of EKAW'94
- [5] Benjamins, Richard - "Problem-Solving Methods for Diagnosis and their Role in Knowledge Acquisition" - Laboratoire de Recherche en Informatique, University of Paris-Sud - International Journal of Expert Systems: Research and Applications, 1995
- [6] Harel, David - "Statecharts: A Visual Formalism for Complex Systems" - Science of Computer Programming 8, pages 231-274 - North-Holland - 1987
- [7] Console, Luca - "Model-based diagnosis history and state of the art" - Monet School <http://www.dimi.uniud.it/dottorato/corsi/aa2000/dague.html> - 2000
- [8] Frydman, C., Le Goc, M, Torres, L., Giambiasi, N. - "The Diagnosis Approach used in SACHEM" - 12th International Workshop on Principles of Diagnosis, p. 63-70 - San Sicario, Italy, 2001
- [9] Basseville, M.; Cordier, M.-O. - "Surveillance et diagnostic de systèmes dynamiques: approches complémentaires du traitement de signal et de l'intelligence artificielle - Publication Interne N° 1004 - IRISA - Mars 1996
- [10] M. Le Goc, C.-C. Thirion - "Using both numerical and symbolic models to create economic value: The SACHEM system example" - Proceedings of the 27th McMaster Symposium on Iron and Steelmaking, pp. 206-218, Ontario, Canada, May 25-27, 1999.

- [11] G. Schreiber, H. Hakkerms, A. Anjewierden, R. de Hoog, N. Shadbolt, W. Van de Velde, B. Wielinga - "Knowledge Engineering and Management - the CommonKADS Methodology" - ISBN 0-262-19300-0 - MIT Press - 2000
- [12] Breuker, J., Van de Velde, W. - "CommonKADS Library for Expertise Modeling" - ISBN 90-5199-164-9 - IOS Press - 1994
- [13] Cauvin, S. - "Un environnement générique à base de connaissances pour la supervision de procédés de raffinage et de pétrochimie" - PhD Thesis - Thèse CNAM - 1995
- [14] GENSYM - "G2 Reference Manual" - Version 3.0 - 1993
- [15] Arzen, K. E. - "A survey of commercial real time expert system environments" - International Symposium on Artificial Intelligence in Real Time Control - Pages 611/ 618 - 1992
- [16] Dousson, C.; Gaborit, P.; Ghallab, M. - "Situation recognition: representation and algorithms" - Proc. of the International Joint Conference on Artificial Intelligence - Pages 166/172 - Chambéry - 1993
- [17] Bousson, K.; Travé, L. - "Putting more numbers in the qualitative simulator CA-EN" - Proceedings 2nd International Conference on Intelligent Systems Engineering - Hamburg - 1994
- [18] Cauvin, S. - "Action plans dynamic application in the alexip knowledge-based system" - Preprints of 2nd Workshop on Comp. Soft. Structures Integrating AI/KBS Systems in Process Control - Lund, Suède - 1994
- [19] Cauvin, S.; Braunschweig, B.; Galtier, P. - "Alexip, expert system coupled with a dynamic simulator for the supervision of the alfabutol process" - Revue de l'Institut Français du Pétrole - Vol. 47, pp.175/182 - 1992
- [20] Bibas, S.; Cordier, M.-O.; Dague, P.; Dousson, C.; Lévy, F.; Rozé, L. - "Scenario generation for telecommunications networks" - Proc. IJCAI Workshop on AI in Dist. Int. Networks - Montréal - 1995
- [21] Bibas, S.; Cordier, M.-O.; Dague, P.; Dousson, C.; Lévy, F.; Rozé, L. - "Alarm driven supervision for telecomm. networks: I-Offline scenario generation" - Annales des Télécomm. - Vol.9/10 pp.493/500 - 1996
- [22] Sampath, M.; Sengupta, R.; Lafortune, S.; Sinnamohideen, K. - "A discrete event system approach to failure diagnosis" - Int. Workshop of Princ. of Diagnosis (DX'94), pp 269/277 - New Paltz, USA - 1994
- [23] Leyval, L.; Gentil, S. - "On line event-based simulation through a causal graph" - IMACS Workshop on Qualitative reasoning and decision support systems, pp. 209/214 - Toulouse - 1991
- [24] Reggia, J.A.; Nau, D. S.; Wang, P. Y. - "Diagnostic expert systems based on a set covering model" - International Journal of Man-Machine Studies - 19(5):437/460 - 1983
- [25] Cayrac, D. - "Diagnostic opérationnel exploitant des connaissances incomplètes" - PhD Thesis - Thèse Université Paul Sabatier - Toulouse - 1995
- [26] Cayrac, D.; Dubois, D.; Prade, H. - "Une méthode pratique de diagnostic de pannes basée sur les modèles avec prise en compte qualitative de l'incertitude" - RFIA'96 pages 320/329 - Rennes - 1996
- [27] Porcheron, M., Ricard, B., Busquet, J.L., Parent, P. - "DIAPO: A case study in applying advances AI techniques to the diagnosis of a complex system" - Proceedings of the European Conference on Artificial Intelligence (ECAI) - Amsterdam - 1994
- [28] Rozé, L. - "Supervision of telecommunications networks: A diagnoser approach" - Proceedings of the International Workshop on Principles of Diagnosis (DX'97), pp. 103/111 - Mont-Saint-Michel - 1997.
- [29] Laborie, P.; Krivine, J.-P. - "Automatic generation of chronicles and its application to alarm processing in power distribution systems" - International Workshop on Principles of Diagnosis (DX'97), pp. 61/68 - Mont-Saint-Michel, France - 1997.
- [30] Buchanan, B.G, Shortlife, E.H.-"Rulebased Expert Systems. The Mycin Experiments of the Stanford Heuristic Programming Project" - Addison Wesley - 1984
- [31] Bonissone, P., Johnson, H. - "Expert System for Diesel Electric Locomotive Repair" - The Journal of FORTH Application and Research, vol. 1, No. 1, pp. 7-16 - September 1983.
- [32] Chandresakan, B., Johnson, T.R., Smith, J.W. - "Task structure analysis for knowledge modeling" - Communications of the ACM, 35(9):124-137 - 1992
- [33] McDermott, J. - "Preliminary steps towards a taxonomy of problem solving methods" - Automating Knowledge Acquisition for Expert Systems, pp. 225-255 - Boston - 1988
- [34] Steels, L. - "Components of Expertise" - AI Magazine, 11(2):28-49 - 1990
- [35] Steels, L. - "The componential framework and its role in reusability" - Second Generation Expert Systems, pp. 273 298 - Berlin, Heidelberg, Germany - Springer Verlag - 1993
- [36] Klinker, G., Bholia, C., Dallemagne, G., Marques, D., McDermott, J. - "Usable and reusable programming constructs" - Knowledge Acquisition, 3/117-136 - 1991
- [37] Puerta, A., Egar, J., Tu, S., Musen, M. - "A multiple method shell for the automatic generation of knowledge acquisition tools" - Knowledge Acquisition, 4/171 196 - 1992
- [38] Angele, J., Fensel, D., Landes, D., Neubert, S., Studer, R. - "Model-based and incremental knowledge engineering: the MIKE approach" - Knowledge Oriented Software Design; IFIP Transactions, A-27 - Amsterdam - 1993
- [39] Pierret-Golbreicht, C. - TASK model, a framework for the design of models of expertise and their operationalization" - Proceedings of the 8th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, pp 37.1-37.22 - Banff, Canada, SRDG Publications - University of Calgary - 1994

Experimental Evaluation of the PLA-Based Permutation-Scheduling

Joanna JĘDRZEJOWICZ

*Institute of Mathematics, Gdańsk University,
Wita Stwosza 57, 80-952 Gdańsk, Poland*

Piotr JĘDRZEJOWICZ

*Department of Information Systems, Gdynia Maritime University,
Morska 83, 81-225 Gdynia, Poland*

Abstract. The paper proposes implementations of the population learning algorithm (PLA) for solving three well-known NP-hard permutation-scheduling problems. PLA is a recently developed method belonging to the class of population-based algorithms and used for solving difficult optimization problems. The first of the discussed problems involves scheduling tasks on a single machine against common due date with earliness and tardiness penalties. The second is known as the permutation flow shop problem. The third one involves scheduling tasks on a single machine with total weighted tardiness as a criterion. To evaluate the proposed implementations computational experiments have been carried. Experiments involved solving available sets of benchmark problems and comparing the results with the optimum or best-known solutions. PLA has found better upper bounds on several benchmark instances.

1. Introduction

Population learning algorithm (PLA), proposed by P. Jędrzejowicz in [12], is a population-based method, which can be applied to solving combinatorial optimization problems. In this paper an experimental evaluation of several implementations of the PLA applied to permutation scheduling is presented.

Permutation scheduling problems are defined as scheduling problems where the solution is a permutation of tasks or jobs to be processed on one or more machines (processors). In the paper three permutation scheduling problems are considered – common due date, permutation flow shop and a single machine weighted tardiness.

Common due date problems with earliness and tardiness penalties are of importance in logistic systems with just-in-time production and delivery. Applications of the problem were discussed by Kanet [13], Hall and Posner [9], Hoogeveen and Van De Velde [10], and later Lee [15], studied properties and complexity of the common due date problem. Both cases i.e. scheduling with restrictive and unrestrictive due dates are known to be NP-hard. It is suspected that the general problem with a restrictive or unrestrictive common due-date is NP-hard in the strong sense. This, however, is still an open question [1].

In a permutation flow shop each machine processes the jobs in the same order [2]. While the machine sequence of all jobs is the same, the problem is to find a job sequence which minimizes the makespan, i.e. the maximum of the completion times of all tasks. It is well known that the problem is NP-hard [7].

Total weighted tardiness problem where jobs are to be processed on a single machine with a view to minimize sum of penalties for being tardy, has been shown to be NP-hard in [4].

In view of the complexity of permutation scheduling problems several authors have proposed various approximation algorithms. For example, several heuristic and metaheuristic algorithms solving the common due dates scheduling problems were proposed in [5], [11], [15]. In case of the permutation flow shop some example heuristic and metaheuristic solutions were discussed by Taillard [20], Tian and others [22], and Reeves[18]. Local search heuristics for the single machine total weighted tardiness scheduling problem were discussed by Crauwels and others [3].

The paper proposes applying a population learning algorithm to obtain solutions to all three problem types instances. It is shown that the approach is effective and promising, moreover, in numerous instances it has been possible to improve previously announced upper bounds for benchmark problems.

The paper is organized as follows. Problem formulation and a short description of the population learning algorithm are given in sections 2 and 3, respectively. Several implementations of the population learning algorithm developed to solve common due date, permutation flow shop and total weighted tardiness scheduling problems are discussed in section 4. Computational experiment results and comparison with benchmark problem solutions are shown in section 5. Conclusions include some suggestions for future research.

2. Problem Formulation

2.1. Scheduling on a single machine against common due date

There is a set of n non-preemptable tasks available at time zero. Tasks are to be processed on a single machine. Processing times of the tasks p_i , $i = 1 \dots n$ are known. There is also a common due date d . Earliness and tardiness of tasks are defined as $E_i = \max\{d - C_i, 0\}$ and $T_i = \max\{C_i - d, 0\}$, respectively ($i = 1 \dots n$) where C_i is the completion time of job i . Penalties per time unit of the task i being early or tardy are a_i and b_i , respectively. The objective is to minimize the sum of earliness and tardiness penalties:

$$\sum_i a_i E_i + \sum_i b_i T_i \rightarrow \min, i = 1 \dots n.$$

It should be also noted that the due date is called unrestrictive if the optimal sequence of tasks can be constructed without considering the value of the due date. Otherwise the common due date is called restrictive. Obviously a common due date for which $d \geq \sum_i p_i$ holds, is unrestrictive.

2.2. Permutation flow shop

Each of n jobs has to be processed on m machines $1 \dots m$ in this order. The processing time of job i on machine j is p_{ij} where the p_{ij} are fixed and nonnegative. At any time, each job can be processed on at most one machine, and each machine can process at most one job. The jobs are available at time 0 and the processing of a job may not be interrupted. In permutation flow shop problem (PFSP) the job order is the same on every machine. The objective is to find a job sequence minimizing schedule makespan (i.e., completion time of the last job).

2.3. Total weighted tardiness

The single-machine total weighted tardiness problem can be stated as follows. Each of n jobs (numbered $1 \dots n$) is to be processed without interruption on a single machine that can handle no more than one job at a time. Job j ($j = 1 \dots n$) becomes available for processing at time zero, requires an uninterrupted positive processing time $p(j)$ on the machine, has a positive weight $w(j)$, and has a due date $d(j)$ by which it should ideally be finished. For a

given processing order of the jobs, the earliest completion time $C(j)$ and the tardiness $T(j) = \max\{C(j) - d(j), 0\}$ of job j ($j = 1 \dots n$) can readily be computed. The problem is to find a processing order of the jobs with minimum total weighted tardiness $\sum_j w(j)T(j)$.

3. Population Learning Algorithm

Population learning algorithm is a population-based method inspired by analogies to a phenomenon of social education processes in which a diminishing number of individuals enter more and more advanced learning stages. In PLA an individual represents a coded solution of the considered problem. Initially, a number of individuals, known as the initial population, is randomly generated.

Once the initial population has been generated, individuals enter the first learning stage. It involves applying some, possibly basic and elementary, improvement schemes. These can be based on some local search procedures. The improved individuals are then evaluated and better ones pass to subsequent stages. A strategy of selecting better or more promising individuals must be defined and duly applied. At the following stages the whole cycle is repeated. Individuals are subject to improvement and learning, either individually or through information exchange, and the selected ones are again promoted to a higher stage with the remaining ones dropped-out from the process. At the final stage the remaining individuals are reviewed and the best represents a solution to the problem at hand.

At different stages of the process, different improvement schemes and learning procedures are applied. These gradually become more and more sophisticated and time consuming as there are less and less individuals to be taught. General idea of the PLA approach is shown in fig. 1.

Although PLA shares some features with evolutionary programs (EP) as discussed, for example, in [16] and greedy randomised adaptive search procedures (GRASP), introduced in [6], it clearly has its own distinctive characteristics. Comparison of all three approaches is shown in table 1.

Table1. Comparison of PLA, EP and GRASP

Features	PLA	EP	GRASP
Processing unit	Population of solutions	Population of solutions	Single solution
Iterative search mode	Selection with decreasing population size	Generation replacement with constant population	Replacement of the current solution
Intensification	Variety of learning and improvement procedures	Some implementations allow for intensification	Greedy constructive heuristic, local search
Information exchange	Different schemes allowed	Crossover	Restricted Candidate List
Diversification	Different schemes allowed	Mutation	Randomised multi-start

PLA shares also some features with other population-based methods. In fact the idea of refining population of solutions during the subsequent computation stages is common to PLA and memetic algorithms [17]. The latter, however, assume a constant population size and a single local search procedure and relay to a greater extend on typical genetic/evolutionary algorithms operators. There are also some similarities between PLA and cultural algorithms where an inheritance process operates at the micro and the micro-evolutionary levels [19].

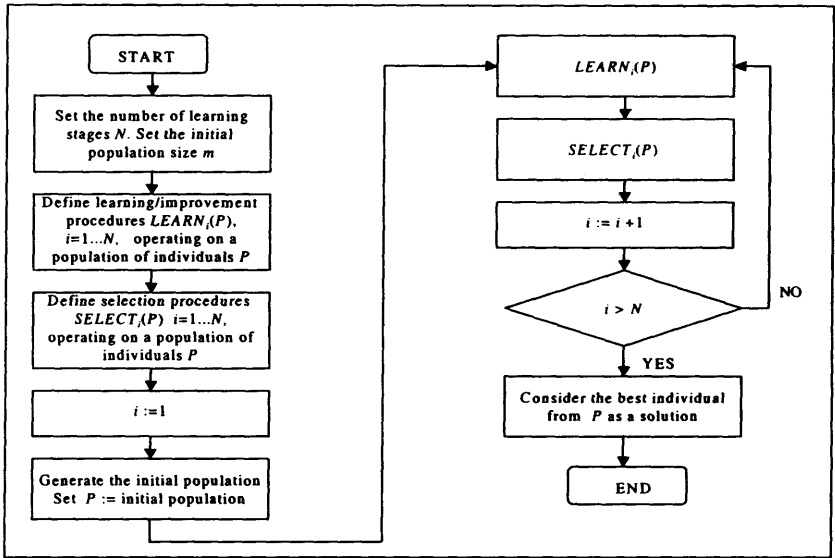


Fig.1. General idea of the population learning algorithm

4. PLA Implementation

To solve permutation scheduling problems three versions of the population learning algorithm, denoted respectively as PLA1, PLA2 and PLA3 have been proposed and implemented. PLA1 has been used to solve the restricted instances of the common due date scheduling problem as well as instances of flow shop and total tardiness problems. PLA2 has been used to solve the unrestricted instances of the common due date problem. Finally, PLA 3 has been designed to solve instances of permutation flow shop and total weighted tardiness problems. The proposed algorithms make use of different learning and improvement procedures, which, in turn, are based on five neighbourhood structures shown in table 2. In what follows x denotes an individual (a permutation of tasks) and $g(x)$ its fitness function.

Table 2. Neighbourhood structures

Notation	Move	Neighbourhood space
$\mathcal{N}_1(x)$	Exchange of two consecutive tasks in x	All possible exchanges
$\mathcal{N}_2(x)$	Exchange of two non-consecutive tasks in x	All possible exchanges
$\mathcal{N}_3(x)$	Finding order of four consecutive tasks in x by enumeration	All fourtuples of consecutive tasks
$\mathcal{N}_4(x)$	Exchange one task with two consecutive tasks in x	All possible exchanges
$\mathcal{N}_5(x)$	Moving a random task in x to another location	All possible moves

All learning and improvement procedures operate on population of individuals P , and perform local search algorithms based on the above defined neighbourhood structures:

```
LEARN (i, P)
for each individual x in P do
    Local_search (i)
end for
```

Local search algorithms used are shown in table 3.

Table 3. Local search algorithms

<i>i</i>	Idea of the local search algorithm
1	Perform all moves from the neighborhood structure $\mathcal{N}_i(x)$; accept moves improving $g(x)$; stop when no further improvements of $g(x)$ are possible
2	Mutate x producing x' {mutation procedure is selected randomly from the two available ones – the two point random exchange or the rotation of all chromosomes between two random points}; perform all moves from the neighborhood structure $\mathcal{N}_i(x')$; accept moves improving $g(x')$; stop when no further improvements of $g(x')$ are possible
3	Repeat k' times { k' is a parameter set at the fine-tuning phase; in the reported experiment $k' = 3$ * initial population size}; perform all moves from the neighborhood structure $\mathcal{N}_i(y)$ and $\mathcal{N}_i(y')$; accept moves improving $g(y)$ and $g(y')$; stop when no further improvements of $g(y)$ and $g(y')$ are possible; adjust P by replacing x and x' with the two best from $\{x, x', y, y'\}$
4	Perform all moves from the neighborhood structure $\mathcal{N}_i(x)$; accept moves improving $g(x)$; stop when no further improvements of $g(x)$ are possible
5	Perform all moves from the neighborhood structure $\mathcal{N}_i(x)$; accept moves improving $g(x)$; stop when no further improvements of $g(x)$ are possible
6	Perform <i>SIMULATED ANNEALING</i> (x)
7	Perform <i>TABU SEARCH</i> (x)

Simulated annealing is a metaheuristic introduced in [14]. Implementation of simulated annealing used within PLA as the 6th local search algorithm is based on random moves from the $\mathcal{N}_i(x)$ neighbourhood structure.

Tabu search is yet another metaheuristic proposed by Glover [8]. General idea of the present implementation using a short term memory (STM) and a long term memory (LTM), is shown in the following pseudo code (STM and LTM are managed according to the reverse elimination method):

TABU SEARCH (x)

begin

$xbest := x$;

$Gbest := g(x)$;

STM, LTM := \emptyset ;

for $j = 1$ **to** it **do** {where it is the number of iterations}

$D_o := -\infty$;

for $i = 1$ **to** lo **do** {where lo is the number of iterations}

Perform *current_move*, which is a random move from $\mathcal{N}_i(x)$ **and** not in STM, producing y from x ;

$D := g(x) - g(y)$;

Update STM;

if $D > D_o$ **then** $D_o = D$; $x' := y$; $best_move := current_move$;

end for

if $g(x') < Gbest$ **then** $Gbest := g(x')$; $xbest := x'$; $x := x'$;

else if $best_move \notin LTM$ **then** $x := x'$;

end if

Update LTM;

Update STM;

end for

$x := xbest$;

end

Now, the structure of the implemented population learning algorithms can be shown as:

PLA1

begin

generate randomly *initial_population*

```

P := initial_population
for i = 1 to 5 do
    LEARN (i, P)
    SELECT (P, s) {SELECT is a procedure in which s percent of individuals
    from P is rejected}
end for
LEARN (6, P)
output the best individual from P

```

end

For the unrestrictive common due date scheduling not only the sequence of tasks but also the starting time of the first task need to be found. To achieve this PLA2 is based on the following assumptions:

- In the course of computation it is assumed that the deadline equals 0 and each task can be scheduled prior to and after time 0.
- Each individual $x = (x_0, x_1, \dots, x_n)$ in the population consists of a sequence of n tasks, $1 \leq x_j \leq n$ for $j = 1 \dots n$ and $x_0 = i \leq n$, which is the number of the last task completed before deadline (x_1, \dots, x_i are completed before time 0 and x_{i+1}, \dots, x_n after time 0), and the starting time of the schedule is $(-\sum_{j=1}^{x_0} p_{x_j})$. In what follows x_0 is called the boundary task.
- In the final step of the computation the real starting time of the first task is computed from: $t_0 = d - \sum_{j=1}^{x_0} p_{x_j}$.

PLA2 uses slightly modified learning and improvement procedures as compared with PLA1. For instance, *LEARN* (3, P) produces an offspring using the following rule: If two individuals x and x' produce y then the boundary task of y equals that of x if the crossover point is greater than the boundary task of x , or the value: (the crossover point + boundary task of $x-1$) otherwise. Same applies to y' .

Further modifications include two new learning and improvement procedures using the following local search procedures:

```

LEARN (8, P)
for each individual x in P do
    Local_search (8) {exchanges two, not necessarily consecutive, tasks, where i (j) is
    less (greater) then the boundary task}
end for
LEARN (9, P)
for each individual x in P do
    Local_search (9) {if the boundary task is less than the number of tasks then it is
    incremented by one}

```

end for

Now PLA2 is defined as follows:

PLA2

begin

```

generate randomly initial_population
P := initial_population
for i = 1,2,3,4,5,8 do
    LEARN (i, P)
    SELECT (P, s)
end for
LEARN (9, P)
output the best individual from P

```

end

Finally, PLA3 has the following structure:

PLA3

begin

 generate randomly *initial_population*

$P := \text{initial_population}$

for $i = 1, 7$ **do**

$LEARN(i, P)$

$SELECT(P, s)$

end for

$LEARN(6, P)$

 output the best individual from P

end

5. Computational experiments

To evaluate the proposed algorithms several computational experiments have been carried on a PC with Pentium III, 850 Mhz processor. The results have been compared with upper bounds or optimal solutions (if known) of several sets of benchmark problems. Table 4 contains short descriptions of benchmark data sets.

Table 4. Benchmark data sets

Problem	Data set description	Source
Common due date	240 problem instances with 20, 50, 100, 200, 500 and 1000 tasks (40 instances for each problem size). Due dates for instances in each problem size group are calculated as: $d = h \sum p_i$ with $h = 0.2, 0.4, 0.6$ and 0.8 , respectively. Problems with $h = 0.6$ and 0.8 are considered as unrestrictive. Upper bounds are provided.	www.wiwi.uni-bielefeld.de/~kistner/Bounds.html (see also [5])
Flow shop	120 problem instances; 10 instances for each combination of number of tasks and number of machines (20-5, 20-10, 20-20, 50-5, 50-10, 50-20, 100-5, 100-10, 100-20, 200-10, 200-20 and 500-20 tasks-machines). Upper bounds are provided.	OR-LIBRARY, http://www.ms.ic.ac.uk/info.html (see also [21])
Weighted tardiness	375 problem instances; 125 instances for each problem size (40, 50 and 100 tasks). Optimal solutions are provided for instances with 40 and 50 tasks. Upper bounds are provided for instances with 100 tasks.	OR-LIBRARY, http://www.ms.ic.ac.uk/info.html

Table 5 shows mean relative errors (MRE) and mean computation times (MCT) for the experiment with common due date scheduling instances. These have been calculated by comparing the best result out of the 3 experimental runs with the upper bounds (that is best up to now known result for benchmark problems). MRE and MCT values in table 5 have been calculated taking the average from 10 instances available for each problem size and due date class. Negative value of the MRE shows, in fact, the relative improvement in comparison to previously established upper bound. In the experiment the initial population size has been set to 2000 and the selection coefficient value s has been set to 0.5.

Table 5. Mean relative errors and MCT for the common due date scheduling

n	PLA1		PLA2		Overall	MCT
	$h = 0.2$	$h = 0.4$	$h = 0.6$	$h = 0.8$		
20	0.00%	0.01%	-0.63%	-0.41%	-0.26%	6.3 s.
50	-0.05%	-0.01%	-0.33%	-0.24%	-0.16%	28.9 s.
100	-0.01%	-0.01%	-0.17%	-0.17%	-0.09%	91.5 s.

200	-0,51%	-0,45%	-0,11%	-0,11%	-0,29%	205.2 s.
500	-5,84%	-5,91%	-2,34%	-2,34%	-4,11%	21.2 m.
1000	-7,42%	-4,39%	-0,03%	-0,04%	-2,97%	43.3 m.
Overall	-2,31%	-1,80%	-0,60%	-0,55%	-1,31%	-

Table 6 shows maximum relative errors (Max RE) and percentages of instances where previously known upper bounds have been improved, as compared with benchmarks.

Table 6. MaxRE and percentages of improved upper bounds – common due date instances

<i>n</i>	Max RE				% of upper bounds improved			
	<i>h</i> = 0.2	<i>h</i> = 0.4	<i>h</i> = 0.6	<i>h</i> = 0.8	<i>h</i> = 0.2	<i>h</i> = 0.4	<i>h</i> = 0.6	<i>h</i> = 0.8
20	no errors	0,1305%	0,1340%	none	0,00%	0,00%	60,00%	50,00%
50	no errors	no errors	0,0211%	0,0182%	10,00%	20,00%	90,00%	90,00%
100	no errors	no errors	0,0015%	0,0015%	100,00%	100,00%	90,00%	90,00%
200	no errors	no errors	no errors	no errors	100,00%	100,00%	100,00%	100,00%
500	no errors	no errors	no errors	no errors	100,00%	100,00%	100,00%	100,00%
1000	no errors	no errors	0,0293%	0,0240%	100,00%	100,00%	80,00%	80,00%

Experiment results prove that PLA1 and PLA2 as applied to common due date scheduling perform remarkably well. Mean relative error for the whole considered population of problem instances is negative and equals -1.31%, which means that the total amount of penalties to be paid as a result of scheduling all 120 problem instances has decreased by 1.31%. Out of 240 problem instances involved in the experiment in 77.5% of cases, that is in 186 instances, it has been possible to find a better upper bound then previously known. The discussed results have been obtained with a reasonable computational effort.

Table 7 shows mean relative errors, maximum relative errors and mean computation times for the experiment with flow shop scheduling instances. These have been calculated by comparing results of a single experimental run of the PLA3 with the upper bounds (that is best up to now known result for benchmark problems). MRE and MCT values in table 7 have been calculated taking the average from 10 instances available for each problem size and number of processors class. The initial population size has been set to 100 and the selection coefficient value *s* has been set to 0.5.

Table 7. Mean and maximum relative errors and mean computation times for the flow shop scheduling

Measure	Processors	<i>n</i> = 20	<i>n</i> = 50	<i>n</i> = 100	<i>n</i> = 200	<i>n</i> = 500
MRE	5	0,0000%	0,0000%	-0,0097%	-	-
				0,1456%		
	10	-0,0073%	0,0566%		0,1471%	-
	20	0,0084%	0,4885%	0,2944%	0,3405%	0,3818%
Max RE	5	0,0000%	0,0000%	0,1329%	-	-
	10	0,0726%	0,6363%	0,4206%	0,3559%	-
	20	0,0860%	0,5777%	0,7425%	0,9606%	0,5594%
MCT	5	2,7m.	9,75m.	28,4m.	-	-
	10	8,5m.	23,0m.	29,1m.	81,0m.	-
	20	15,2m.	33,4m.	38,8m.	140,0m	295m.

Experiment results prove that PLA3 as applied to permutation flow shop scheduling perform quite well. Mean relative error for the whole considered population of problem instances equals only 0.1776% and this has been obtained in a single run. Out of 120 problem instances involved in the experiment in 10% of cases, that is in 12 instances, it has been possible to find a better upper bound then previously known. Considering the research

effort invested during the last 20 years into finding algorithms solving flow shop problems the quality of PLA3 is impressive. The discussed results, however, required a substantial computational effort.

Table 8 shows mean relative errors, maximum relative errors and mean computation times for the total tardiness scheduling. These have been calculated by comparing results of a single experimental run of the PLA1 and PLA3 with the optimal solutions in case of 40 and 50 tasks and upper bounds in case of 100 tasks instances. MRE and MCT values in table 8 have been calculated taking the average from 125 instances available for each problem size. The initial population size has been set to 1000 for the PLA1 and 100 for the PLA3, and the selection coefficient value s has been set to 0.5.

Table 8. MRE, Max RE and MCT for the total tardiness scheduling problem

n	PLA1			PLA3		
	MRE	MaxRE	MCT	MRE	Max RE	MCT
40	0,0091%	0,8129%	11,6s.	0,0000%	0,0000%	4,03m.
50	0,1206%	2,1828%	18,6s.	0,0028%	0,1617%	4,18m.
100	0,1549%	4,8972%	1,21m.	0,0006%	0,0404%	20,43m.

Experiment results prove that PLA1 and PLA3 as applied to total tardiness scheduling perform very well. In case of PLA1 single run, mean relative error for the whole considered population of problem instances equals only 0.11%. The same measure in case of PLA3 has value of 0.001%. However, in case of the PLA3 computational effort involved is substantially higher.

Best schedules improving previously known upper bounds that have been obtained during the reported experiments are available at <http://manta.univ.gda.pl/~jj/pla.txt>.

6. Conclusions

Considering the results of the experiment involving the proposed implementations of the population learning algorithm and three computationally difficult permutation scheduling problems, the following conclusions can be drawn:

- Population learning algorithms can be considered as an effective metaheuristic for solving difficult scheduling problems. Experiments carried out have contributed to improving 77,5% of upper bounds out of 240 benchmark instances solved in case of the scheduling against common due date problem and 10% of upper bounds out of 120 benchmark instances in case of the flow shop problem. In case of 375 benchmark instances for the total tardiness scheduling PLA produces mean relative error of 0,001% as compared with optimal solutions.
- PLA seems to be an excellent vehicle for implementing hybrid algorithms based on computational intelligence technologies.
- Critical for the successful application of the PLA approach is a fine-tuning phase where a proper balance between computation time and solution quality has to be found. Since there are several parameters, which can be controlled by the user (population size, selection criterion, number of iterations for the embedded tabu search and simulated annealing procedures, etc.) further research is needed to identify interactions between these parameters and to establish methodology for setting their values.
- Whereas quality of the solutions obtained using PLA algorithms is satisfactory, there is still a lot of room for improvements with respect to computational effort required.
- Future research should concentrate on parallel PLA schemes with a view of increasing quality of results or decreasing computation time or achieving both goals simultaneously.

- It can be suspected that the quality of upper bounds for benchmark instances is much higher in case of the flow shop problem than in case of scheduling on a single machine against common due date problem.

7. Acknowledgement

The research has been supported by the KBN grant no. 8T11F02019.

References

- [1] Biskup D., M. Feldmann: Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates, *Computers & Operations Research*, 28 (2001) 787-801.
- [2] Błażewicz J., K.H.Ecker, E.Pesch, G.Schmidt, J.Węglarz: *Scheduling Computer and Manufacturing Processes*, Springer, Berlin, 1996.
- [3] Crauwels H.A.J., C.N. Potts and L.N. Van Wassenhove: Local search heuristics for the single machine total weighted tardiness scheduling problem, *Inform Journal on Computing*, 10(1998) 341-350.
- [4] Du J., J.Y.Leung: Minimizing total tardiness on one machine is NP-hard, *Mathematics of Operations Research*, 15(1990) 483-495.
- [5] Feldmann M., D. Biskup: Single-machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches, *Discussion Paper No 425*, University of Bielefeld, Bielefeld, 1999.
- [6] Feo T.A., M.G.C. Resende: A probabilistic heuristic for computationally difficult set covering problems, *Operations Research Letters*, 8(1989) 706-712.
- [7] Garey M.R., D.S.Johnson, R.Sethi: The complexity of flowshop and jobshop scheduling, *Math. Oper. Res.*, 1 (1976) 117-129.
- [8] Glover F.: Tabu search: a tutorial, *Interfaces*, 20 (1990) 74-94.
- [9] Hall N.G., M. E. Posner: Earliness-tardiness scheduling problems, I: weighted deviation of completion times about a common due date, *Operations Research*, 39 (1991) 836-846.
- [10] Hoogeveen J.A., S. L. Van De Velde: Scheduling around a small common due date, *European Journal of Operational Research*, 55 (1991) 237-242.
- [11] James R.J.W.: Using tabu search to solve the common due date early/tardy machine scheduling problem, *Computers & Operations Research*, 24 (1997) 199-208.
- [12] Jędrzejowicz P.: Social Learning Algorithm as a Tool for Solving Some Difficult Scheduling Problems, *Foundation of Computing and Decision Sciences*, 24 (1999) 51-66.
- [13] Kanet J.J.: Minimizing the average deviation of job completion times about a common due date, *Naval Research Logistics Quarterly*, 28 (1981) 643-651.
- [14] Kirkpatrick S., C.D.Gelatt Jr., M.P.Vecchi: Optimization by simulated annealing, *Science*, 220 (1983) 671-680.
- [15] Lee C.Y., S. J. Kim: Parallel genetic algorithms for the earliness-tardiness job scheduling problem with general penalty weights, *Computers & Industrial Engineering*, 28 (1995) 231-243.
- [16] Michalewicz Z.: *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Berlin, 1992.
- [17] Moscato P.: Memetic Algorithms: A short introduction. In: D.Corne, M.Dorigo, F.Glover (Eds.), *New Ideas in Optimization*, McGraw-Hill, New York, 1999, 219-234
- [18] Reeves C.: A genetic algorithm for flowshop sequencing, *Computers and Operations Research*, 22 (1995) 5-13.
- [19] Reynolds R.G. : An Introduction to Cultural Algorithms. In: A.V. Sebald, L.J. Fogel (Eds.), *Proceedings of the Third Annual Conference on Evolutionary Programming*, World Scientific , River Edge, 1994, 131-139.
- [20] Taillard E.: Some efficient heuristic methods for the flow shop sequencing problem, *European Journal of Operational Research*, 47 (1990) 65-74.
- [21] Taillard E.: Benchmarks for basic scheduling instances, *European Journal of Operational Research*, 64 (1993) 278-285.
- [22] Tian P., J.Ma, D.M.Zhang: Application of the simulated annealing algorithm to the combinatorial optimisation problem with permutation property: An investigation of generation mechanism, *European Journal of Operational Research*, 118(1999) 81-94.

A Study of K -Nearest Neighbour as an Imputation Method

Gustavo E. A. P. A. Batista and Maria Carolina Monard
University of São Paulo - USP

Institute of Mathematics and Computer Science - ICMC

Department of Computer Science and Statistics - SCE

Laboratory of Computational Intelligence - LABIC

P. O. Box 668, 13560-970 - São Carlos, SP, Brazil

{gbatista, mcmonard}@icmc.usp.br

Abstract. Data quality is a major concern in Machine Learning and other correlated areas such as Knowledge Discovery from Databases (KDD). As most Machine Learning algorithms induce knowledge strictly from data, the quality of the knowledge extracted is largely determined by the quality of the underlying data. One relevant problem in data quality is the presence of missing data. Despite the frequent occurrence of missing data, many Machine Learning algorithms handle missing data in a rather naive way. Missing data treatment should be carefully thought, otherwise bias might be introduced into the knowledge induced. In this work, we analyse the use of the k -nearest neighbour as an imputation method. Imputation is a term that denotes a procedure that replaces the missing values in a data set by some plausible values. Our analysis indicates that missing data imputation based on the k -nearest neighbour algorithm can outperform the internal methods used by C4.5 and CN2 to treat missing data.

1 Introduction

Data quality is a major concern in Machine Learning and other correlated areas such as Knowledge Discovery from Databases (KDD). As most Machine Learning algorithms induce knowledge strictly from data, the quality of the knowledge extracted is largely determined by the quality of the underlying data.

One relevant problem in data quality is the presence of missing data. Missing data may have different sources such as death of patients, equipment malfunctions, refusal of respondents to answer certain questions, and so on.

Despite the frequent occurrence of missing data, many Machine Learning algorithms handle missing data in a rather naive way. Missing data treatment should be carefully thought, otherwise bias might be introduced into the knowledge induced.

Imputation is a term that denotes a procedure that replaces the missing values in a data set by some plausible values. One advantage of this approach is that the missing data treatment is independent of the learning algorithm used. This allows the user to select the most suitable method for each situation.

The objective of this work is to analyse the performance of the k -nearest neighbour as an imputation method for missing data. The performance of this method is compared to the performance of two well known Machine Learning algorithm: CN2 [4] and C4.5 [12].

This work is organized as follows: Section 2 describes the taxonomy proposed by [10] to classify the degree of randomness of the missing data in a data set; Section 3 surveys the most used methods for missing data treatment; Section 4 is dedicated to a specific class of missing data treatment methods: imputation; Section 5 presents the k -nearest neighbour as an imputation method for treating missing values; Section 6 describes how the Machine Learning algorithms C4.5 and CN2 treat missing data internally; Section 7 performs a comparative study of the k -nearest neighbour algorithm as an imputation method with the internal methods used by C4.5 and CN2 to treat missing data; finally, Section 8 presents the conclusions of this work.

2 Randomness of Missing Data

Missing data randomness can be divided into three classes, as proposed by [10]:

1. *Missing completely at random (MCAR)*. This is the highest level of randomness. It occurs when the probability of an instance (case) having a missing value for an attribute does not depend on either the known values or the missing data. In this level of randomness, any missing data treatment method can be applied without risk of introducing bias on the data;
2. *Missing at random (MAR)*. When the probability of an instance having a missing value for an attribute may depend on the known values, but not on the value of the missing data itself;
3. *Not missing at random (NMAR)*. When the probability of an instance having a missing value for an attribute could depend on the value of that attribute.

3 Methods for Treating Missing Data

There are several methods for treating missing data available in the literature. Many of these methods, such as case substitution, were developed for dealing with missing data in sample surveys, and have some drawbacks when applied to the Data Mining context. Other methods, such as replacement of missing values by the attribute mean or mode, are very naive and should be carefully used to avoid insertion of bias.

In a general way, missing data treatment methods can be divided into three categories, as proposed in [10]:

1. *Ignoring and discarding data*. There are two main ways to discard data with missing values. The first one is known as *complete case analysis*, it is available in all statistical programs and is the default method in many programs. This method consists of discarding all instances (cases) with missing data. The second method is known as *discarding instances and/or attributes*. This method consists of determining the extent of missing data on each instance and attribute, and delete the instances and/or attributes with high levels of missing data. Before deleting any attribute, it is

necessary to evaluate its relevance to the analysis. Unfortunately, relevant attributes should be kept even with a high degree of missing values. Both methods, complete case analysis and discarding instances and/or attributes, should be applied only if missing data are MCAR, because missing data that are not MCAR have non-random elements that can bias the results;

2. *Parameter estimation.* Maximum likelihood procedures are used to estimate the parameters of a model defined for the complete data. Maximum likelihood procedures that use variants of the Expectation-Maximization algorithm [5] can handle parameter estimation in the presence of missing data;
3. *Imputation.* Imputation is a class of procedures that aims to fill in the missing values with estimated ones. The objective is to employ known relationships that can be identified in the valid values of the data set to assist in estimating the missing values. This papers focus on imputation of missing data. More details about this class of methods are described next.

4 Imputation Methods

Imputation methods involve replacing missing values with estimated ones based on some information available in the data set. There are many options varying from naive methods like mean imputation to some more robust methods based on relationships among attributes.

This section surveys some widely used imputation methods, although others forms of imputation are available.

1. *Case substitution.* This method is typically used in sample surveys. One instance with missing data (for example, a person that cannot be contacted) is replaced by another nonsampled instance;
2. *Mean and mode.* One of the most frequently used methods. This method consists of replacing the missing data for a given attribute by the mean (quantitative attribute) or mode (qualitative attribute) of all known values of that attribute;
3. *Hot deck and cold deck.* In the hot deck method, a missing attribute value is filled in with a value from an estimated distribution for the missing value from the current data. Hot deck is typically implemented into two stages. In the first stage, the data are partitioned into clusters. And, in the second stage, each instance with missing data is associated with one cluster. The complete cases in a cluster are used to fill in the missing values. This can be done by calculating the mean or mode of the attribute within a cluster. Cold deck imputation is similar to hot deck but the data source must be other than the current data source;
4. *Prediction model.* Prediction models are sophisticated procedures for handling missing data. These methods consist of creating a predictive model to estimate values that will substitute the missing data. The attribute with missing data is used as class-attribute, and the remaining attributes are used as input for the predictive

model. An important argument in favour of this approach is that, frequently, attributes have relationships (correlations) among themselves. In this way, those correlations could be used to create a predictive model for classification or regression (depending on the attribute type with missing data, being, respectively, nominal or continuous). Some of these relationships among the attributes may be maintained if they were captured by the predictive model. An important drawback of this approach is that the model estimated values are usually more well-behaved than the true values would be, *i.e.*, since the missing values are predicted from a set of attributes, the predicted values are likely to be more consistent with this set of attributes than the true (not known) value would be. A second drawback is the requirement for correlation among the attributes. If there are no relationships among one or more attributes in the data set and the attribute with missing data, then the model will not be precise to estimate the missing values.

5 Imputation with k -Nearest Neighbour

In this work we propose the use of k -nearest neighbour algorithm to estimate and substitute missing data. The main benefits of this approach are:

- k -nearest neighbour can predict both discrete attributes (the most frequent value among the k nearest neighbours) and continuous attributes (the mean among the k nearest neighbours);
- There is no necessity for creating a predictive model for each attribute with missing data. Actually, the k -nearest neighbour does not create explicit models (like a decision tree or a set of rules), once the data set is used as a “lazy” model. Thus, the k -nearest neighbour can be easily adapted to work with any attribute as class, by just modifying which attributes will be considered in the distance metric. Also, this approach can easily treat examples with multiple missing values.

The main drawback of this approach is:

- Whenever the k -nearest neighbour looks for the most similar instances, the algorithm searches through all the data set. This limitation can be very critical for KDD, since this research area has, as one of its main objectives, the analysis of large databases. Several works that aim to solve this limitation can be found in the literature. One method is the creation of a reduced training set for the k -nearest neighbour composed only by prototypical examples [13]. In this work we use an access method called M-tree [3], that we have implemented in our k -nearest neighbour algorithm. M-trees can organize and search data sets based on a generic metric space. M-trees can drastically reduce the number of distance computations in similarity queries.

6 How C4.5 and CN2 Treat Missing Data

C4.5 and CN2 are two well known Machine Learning Algorithms that induce propositional concepts: decision trees and rules, respectively. These algorithms were selected because they are considered two of the best Machine Learning algorithms. C4.5 seems to have a good internal algorithm to treat missing values, since a recent comparative

study, with other simple methods to treat missing values, concluded that it was one of the best methods [6]. On the other hand, CN2 seems to use a rather simple method to treat missing data.

C4.5 uses a probabilistic approach to handle missing data. Missing values can be present in any attribute, except the class attribute, in training and test files.

Given a training set, T , C4.5 finds a suitable test, based on a single attribute, that has one or more mutually exclusive outcomes O_1, O_2, \dots, O_n . T is partitioned into subsets T_1, T_2, \dots, T_n , where T_i contains all the instances in T that satisfy the test with outcome O_i . The same algorithm is applied to each subset T_i until a stop criteria is applied.

C4.5 uses the *information gain ratio* measure to choose a good test to partition the instances. If there exist missing values in an attribute X , C4.5 uses the subset with all known values of X to calculate the information gain.

Once a test based on an attribute X is chosen, C4.5 uses a probabilistic approach to partition the instances with missing values in X . When an instance in T with known value is assigned to a subset T_i , this indicates that the probability of that instance belonging to subset T_i is 1 and to all other subsets is 0. When the value is not known, only a weaker probabilistic statement can be made. C4.5 associates to each instance in T_i a *weight* representing the probability of that instance belonging to T_i . If the instance has a known value, and satisfies the test with outcome O_i , then this instance is assigned to T_i with weight 1; if the instance has an unknown value, this instance is assigned to all partitions with different weights for each one. The weight for the partition T_i is the probability of that instance belongs to T_i . This probability is estimated as the sum of the weights of instances in T known to satisfy the test with outcome O_i , divided by the sum of weights of the cases in T with known values on the attribute X .

The CN2 algorithm uses a rather simple imputation method to treat missing data. Every missing value is filled in with its attribute most common known value, before calculating the entropy measure [4].

7 Experimental Analysis

The main objective of the experiments conducted in this work is to evaluate the efficiency of the k -nearest neighbour algorithm as an imputation method to treat missing data, comparing its performance with the performance obtained by the internal algorithms used by C4.5 and CN2 to learn with missing data.

In our experiments, missing values were artificially implanted, in different rates and attributes, into the data sets. The performance of all three missing data treatments are compared using cross-validation estimated error rates. In particular, we are interested in analysing the behaviour of these treatments when the amount of missing data is high since some researchers have reported to find databases where more than 50% of the data were missing, for instance [8].

The experiments were carried using three data sets from UCI [11]: Bupa, Cmc and Pima. We chose these three data sets because they have no missing values. The main reason for this choice is that we want to have total control over the missing data in the data set. For instance, we would like that the test sets do not have any missing data. If some test set has missing data, then the inducer's ability to classify missing data

properly may influence on the result. This influence is undesirable since the objective of this work is to analyse the viability of the k -nearest neighbour as imputation method for missing data and the inducer learning ability when missing values are present. Table 1 summarizes the data sets employed in this study. It shows, for each data set, the number of instances (#Instances), number and percentage of duplicate (appearing more than once) or conflicting (same attribute-value but different class attribute) instances, number of attributes (#Attributes) quantitative and qualitative, class attribute distribution and the majority class error. These information has been obtained using the MLC++ *info* utility [7].

Data set	# Instances	#Duplicate or conflicting (%)	#Attributes (quantit., quali.)	Class	Class %	Majority Error
bupa	345	4 (1.16%)	6 (6,0)	1	42.03%	42.03% on value 2
				2	57.97%	
cmc	1473	115 (7.81%)	9 (2,7)	1	42.70%	57.30% on value 1
				2	22.61%	
				3	34.69%	
pima	769	1 (0.13%)	8 (8,0)	0	65.02%	34.98% on value 0
				1	34.98%	

Table 1: Data sets Summary Descriptions

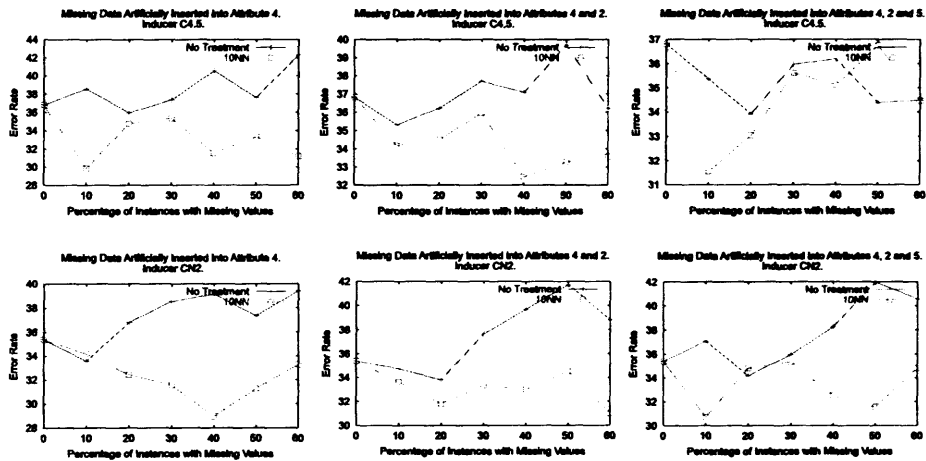


Figure 1: Comparative results for the Bupa data set.

Initially, the original data set is partitioned into 10 pairs of training and test sets through the application of 10-fold cross validation resampling method. Then, missing values are inserted into the training set. Four copies of this training set are used, two are given to C4.5 and CN2 without any missing data treatment. In the other two, the k -nearest neighbour is used to estimate and substitute missing values. After the treatment of missing data, the training sets are given to C4.5 and CN2. All classifiers, *i.e.* the two induced with untreated data and the other two induced with treated data, are used to classify the test set. At the end of 10 iterations, we can estimate the true error rate by calculating the mean of the error rates of each iteration. Finally, the performances of C4.5 and CN2 allied to the missing data treatment method can be analysed and

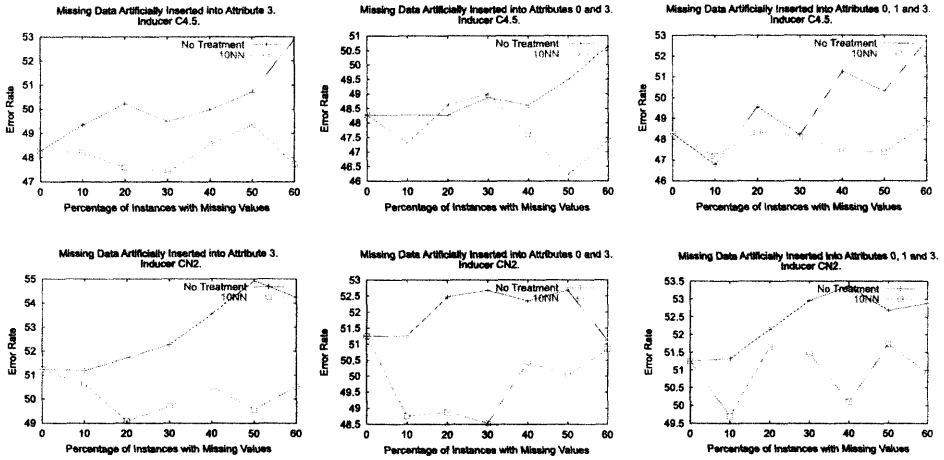


Figure 2: Comparative results for the Cmc data set.

compared to the performance of the method used internally by C4.5 and CN2 to learn when missing values are present.

In order to insert missing data into the training sets, some attributes have to be chosen, and some of their values modified to unknown. Which attributes will be chosen and how many of their values will be modified to unknown is an important decision. It is easy to see that the most representative attributes of the data set are a sensible choice for the attributes that should have their values modified to unknown. Otherwise, the analysis may be compromised by treating non-representative attributes that will not be incorporated into the classifier by the learning system. Since finding the most representative attributes of a data set is not a trivial task, we used the results of [9] to select the three most relevant attributes according to several feature subset selection method such as wrapper and filter.

There is no assurance that these attributes will be incorporated by C4.5 and CN2 into the classifiers induced in the experiments. The existence of an attribute with similar information (high correlation) with one of the selected attributes, can make the inducers choose not to use the selected attribute.

Related to the amount of missing data to be inserted into the training sets, we want to analyse the behaviour of the methods with different amounts of missing data. In this way, missing data was inserted completely at random (MCAR) in the following percentages: 10%, 20%, 30%, 40%, 50% and 60% of the total of instances.

The experiments were done with missing data inserted into one, two and all three attributes selected as the most representative. The missing values were replaced by estimated values using 1, 3, 5, 10, 20, 30, 50 and 100 nearest neighbours. Unfortunately, because of lack of space, only results with 10-nearest neighbour, identified as 10-NNI, will be showed in this work.

Considering the results shown in Figure 1, the performance of 10-NNI is superior to the performance of both C4.5 and CN2 algorithms for the Bupa data set. The only situation that C4.5 is competitive to 10-NNI is when missing values are inserted into the attributes 4, 2 and 5.

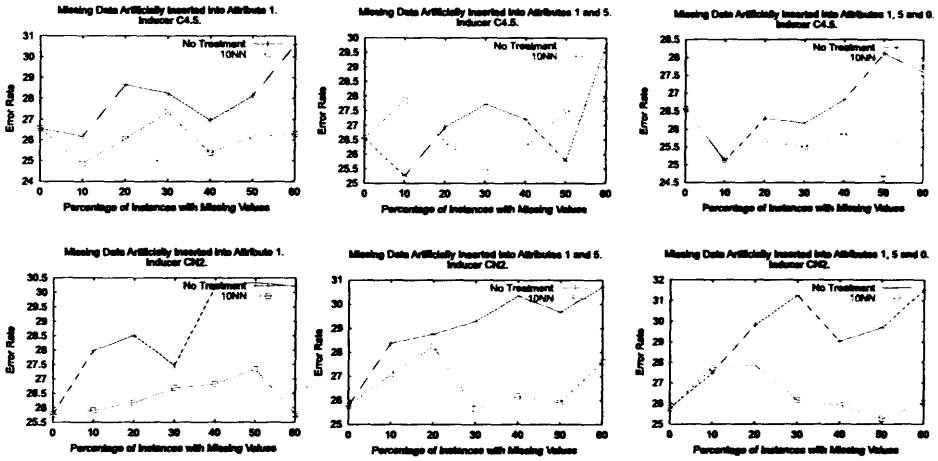


Figure 3: Comparative results for the Pima data set.

Similar results are shown in Figure 2. The performance of the imputation method 10-NNI is superior to the performance obtained, without missing data treatment, for both C4.5 and CN2 in the Cmc data set.

Finally, Figure 3 shows the comparative results for the Pima data set. In this data set, the method 10-NNI had a slightly superior performance compared with C4.5, and a superior performance compared with CN2.

Table 2 shows some numerical results of the graphs presented in Figures 1, 2 and 3. In this table are showed the error rates, standard deviations and 10-fold cross validation paired t-test results. More detailed results can be seen in [1].

It is important to say that the internal methods to treat missing data of C4.5 and CN2 had lower error rates compared to 10-NNI only in 11 of 108 measurements (8 for C4.5 and 3 for CN2). In none of these 11 measurements the internal methods had a statistically significant difference. On the other hand, 10-NNI had statistically significant difference in 35 measurements (13 of these measurements were high significant).

8 Conclusion

The main objective of this work is to present some results of a research that aims to analyse the benefits and drawbacks of missing data treatment methods [2]. In this work, we analyse the behaviour of three methods for missing data treatment: the 10-NNI method using a k -nearest neighbour algorithm for missing data imputation; and the internal algorithms used by C4.5 and CN2 to treat missing data. These methods were analysed inserting different percentages of missing data into different attributes. The results are very promising. The 10-NNI method provide very good results, even when the training sets had a large amount of missing data.

		C4.5				CN2			
	??	No Imputation	t-test	10-NN	No Imputation	t-test	10-NN		
Bupa	4	0%	36.82 ± 2.69	-	-	35.39 ± 2.47	-	-	-
		10%	38.56 ± 1.74	↑ -3.01	29.87 ± 1.76	33.58 ± 1.94	0.22	34.19 ± 1.45	
		20%	35.95 ± 1.24	-0.67	34.78 ± 2.43	36.82 ± 0.96	↑ -3.51	32.45 ± 0.95	
		30%	37.36 ± 1.89	-0.72	35.36 ± 2.71	38.53 ± 2.16	↑ -4.02	31.56 ± 2.71	
		40%	40.56 ± 2.05	↑ -3.09	31.55 ± 1.86	39.13 ± 1.09	↑ -6.21	28.96 ± 2.24	
		50%	37.62 ± 2.35	-1.16	33.34 ± 2.54	37.35 ± 2.74	-2.02	31.28 ± 1.91	
		60%	42.31 ± 2.11	↑ -2.62	31.22 ± 3.33	39.41 ± 1.20	↑ -2.38	33.29 ± 2.64	
	4 and 2	0%	36.82 ± 2.69	-	-	35.39 ± 2.47	-	-	-
		10%	35.32 ± 2.36	-0.52	34.18 ± 1.72	34.75 ± 2.01	-0.57	33.63 ± 1.77	
		20%	36.22 ± 2.18	-0.57	34.51 ± 2.16	33.81 ± 3.23	-0.64	31.81 ± 2.65	
		30%	37.70 ± 2.40	-0.90	35.96 ± 2.05	37.66 ± 1.48	↑ -2.65	33.34 ± 1.88	
		40%	37.08 ± 1.42	↑ -2.51	32.45 ± 1.09	39.67 ± 1.98	-2.25	33.02 ± 2.44	
		50%	39.71 ± 2.76	-1.85	33.28 ± 3.07	41.72 ± 1.38	↑ -3.04	34.51 ± 2.40	
		60%	36.21 ± 1.84	-1.09	33.57 ± 2.38	38.81 ± 1.58	↑ -3.90	31.01 ± 1.48	
	4, 2 and 5	0%	36.82 ± 2.69	-	-	35.39 ± 2.47	-	-	-
		10%	35.36 ± 1.76	-1.51	31.56 ± 2.44	37.09 ± 2.55	↑ -2.45	30.71 ± 2.47	
		20%	33.92 ± 2.07	-0.36	33.05 ± 2.09	34.18 ± 2.03	0.38	34.81 ± 1.49	
		30%	35.97 ± 2.90	-0.08	35.61 ± 3.00	35.94 ± 2.14	-0.24	35.35 ± 1.39	
		40%	36.19 ± 2.39	-0.38	35.11 ± 2.14	38.25 ± 1.49	↑ -3.09	32.49 ± 1.20	
		50%	34.39 ± 2.84	0.97	36.75 ± 2.12	41.97 ± 1.58	↑ -5.34	31.56 ± 1.58	
		60%	34.48 ± 1.77	-0.00	34.47 ± 3.02	40.56 ± 1.88	-2.05	34.82 ± 2.04	
Cmc	3	0%	48.27 ± 0.83	-	-	51.25 ± 0.80	-	-	-
		10%	49.35 ± 1.14	-1.27	48.20 ± 1.16	51.19 ± 1.51	-0.48	50.64 ± 1.22	
		20%	50.23 ± 1.12	↑ -2.31	47.59 ± 0.98	51.73 ± 1.17	↑ -2.61	49.08 ± 0.95	
		30%	49.49 ± 0.95	-2.16	47.39 ± 1.48	52.27 ± 0.94	-1.96	49.70 ± 1.71	
		40%	49.97 ± 0.87	-1.58	48.54 ± 1.12	53.56 ± 1.47	↑ -2.28	50.51 ± 1.11	
		50%	50.71 ± 1.11	-1.14	49.36 ± 0.91	54.92 ± 0.95	↑ -4.18	49.56 ± 1.74	
		60%	52.88 ± 1.25	↑ -7.08	47.73 ± 0.95	54.24 ± 1.31	↑ -2.63	50.51 ± 1.12	
	3 and 0	0%	48.27 ± 0.83	-	-	51.25 ± 0.80	-	-	-
		10%	48.27 ± 0.67	-1.10	47.32 ± 1.30	51.26 ± 0.80	-2.16	48.75 ± 1.42	
		20%	48.27 ± 0.99	0.30	48.61 ± 1.30	52.48 ± 1.51	↑ -2.84	48.88 ± 1.46	
		30%	48.88 ± 1.40	0.09	49.02 ± 1.36	52.68 ± 0.91	↑ -2.77	48.54 ± 1.34	
		40%	48.61 ± 1.20	-0.76	47.59 ± 1.53	52.35 ± 1.10	↑ -2.88	50.44 ± 1.09	
		50%	49.49 ± 0.84	↑ -3.16	46.23 ± 1.06	52.68 ± 0.81	-1.60	50.03 ± 1.76	
		60%	50.64 ± 1.16	-1.62	47.39 ± 1.87	51.12 ± 1.53	-0.28	50.85 ± 1.49	
	3, 0 and 1	0%	48.27 ± 0.83	-	-	51.25 ± 0.80	-	-	-
		10%	46.78 ± 1.46	0.29	47.18 ± 1.19	51.32 ± 1.19	-1.13	49.70 ± 1.61	
		20%	49.56 ± 1.34	-1.75	48.34 ± 1.29	52.14 ± 1.04	-0.39	51.66 ± 1.06	
		30%	48.20 ± 1.19	-0.05	48.13 ± 1.51	52.95 ± 1.25	-1.55	51.46 ± 1.15	
		40%	51.26 ± 1.33	↑ -3.18	47.45 ± 1.46	53.36 ± 1.23	↑ -2.91	50.10 ± 1.49	
		50%	50.31 ± 1.23	↑ -2.52	47.38 ± 1.74	52.68 ± 1.02	-0.55	51.73 ± 1.82	
		60%	52.75 ± 1.16	↑ -3.49	48.75 ± 1.86	52.88 ± 0.76	-1.32	50.92 ± 1.35	
Pima	1	0%	26.56 ± 1.16	-	-	25.77 ± 1.12	-	-	-
		10%	26.17 ± 1.03	-1.06	24.86 ± 0.88	27.99 ± 0.98	-2.23	25.91 ± 0.86	
		20%	28.65 ± 1.15	-1.48	26.04 ± 1.68	28.51 ± 1.06	-1.80	26.18 ± 0.78	
		30%	28.25 ± 1.85	-0.46	27.35 ± 1.03	27.47 ± 1.11	-0.41	26.69 ± 1.61	
		40%	26.95 ± 1.67	-0.93	25.38 ± 1.15	30.21 ± 1.08	-2.00	26.82 ± 0.98	
		50%	28.11 ± 1.14	-1.10	26.17 ± 1.11	30.34 ± 1.21	-1.54	27.35 ± 1.47	
		60%	30.59 ± 1.13	-2.21	26.29 ± 1.90	30.21 ± 1.28	-2.21	25.78 ± 1.33	
	1 and 5	0%	26.56 ± 1.16	-	-	25.77 ± 1.12	-	-	-
		10%	25.25 ± 1.10	2.00	27.86 ± 1.15	28.38 ± 0.87	-1.32	27.08 ± 0.98	
		20%	26.94 ± 1.22	-0.39	26.43 ± 1.08	28.76 ± 1.51	-0.32	28.25 ± 1.09	
		30%	27.73 ± 1.60	-1.26	25.39 ± 0.81	29.30 ± 1.23	↑ -2.45	25.65 ± 1.13	
		40%	27.21 ± 1.45	-0.51	26.29 ± 1.69	30.34 ± 1.59	↑ -2.56	26.17 ± 1.07	
		50%	25.78 ± 1.13	1.39	27.46 ± 1.16	29.68 ± 1.58	-2.02	25.91 ± 1.08	
		60%	29.81 ± 1.43	-1.18	27.85 ± 1.51	30.72 ± 1.47	-1.18	27.60 ± 1.47	
	1, 5 and 0	0%	26.56 ± 1.16	-	-	25.77 ± 1.12	-	-	-
		10%	25.11 ± 1.70	0.01	25.13 ± 0.90	27.48 ± 1.00	0.21	27.73 ± 0.68	
		20%	26.30 ± 1.01	-0.66	25.65 ± 1.35	29.82 ± 0.82	-1.48	27.87 ± 1.26	
		30%	26.17 ± 1.35	-0.38	25.51 ± 1.75	31.25 ± 0.89	↑ -3.55	26.17 ± 1.32	
		40%	26.82 ± 1.28	-0.67	25.91 ± 1.44	29.03 ± 0.90	↑ -3.67	25.92 ± 1.32	
		50%	28.11 ± 1.32	↑ -3.41	24.61 ± 1.16	29.69 ± 0.41	↑ -7.98	25.26 ± 0.68	
		60%	27.60 ± 1.05	0.19	27.86 ± 1.55	31.51 ± 1.17	↑ -3.05	26.05 ± 0.86	

Table 2: Comparative results for all data sets. “↑” means that the difference is statistically significant (95% confidence level). “↑↑” means that the difference is highly significant (99% confidence level).

In future works, the missing data treatment methods will be analyzed in other data sets. Furthermore, in this work missing values are being inserted completely at random (MCAR). In a future work, we will analyze the behaviour of these methods when missing values are not randomly distributed. In this case, there exist the possibility of invalid knowledge be created by the inducer. For an effective analysis, we will have to inspect not only the error rate, but also, the quality of the knowledge induced by the learning system.

Acknowledgements. This research is partially supported by Brazilian Research Councils CAPES and FINEP.

References

- [1] G. E. A. P. A. Batista and M. C. Monard. K-Nearest Neighbour as Imputation Method: Experimental Results (in print). Technical report, ICMC-USP, 2002. ISSN-0103-2569.
- [2] Gustavo E. A. P. A. Batista. Data Pre-processing for Supervised Learning (in Portuguese). Phd Qualifying Exam, ICMC-USP, March 2000.
- [3] Paolo Ciaccia, Marco Patella, and Pavel Zezula. M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. In *The VLDB Journal*, pages 426–435, 1997.
- [4] P. Clark and T. Niblett. The CN2 Induction Algorithm. *Machine Learning*, 3(4):261–283, 1989.
- [5] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm (with Discussion). *Journal of Royal Statistical Society*, B39:1–38, 1977.
- [6] J. W. Grzymala-Busse and M. Hu. A Comparison of Several Approaches to Missing Attribute Values in Data Mining. In *Proceedings of the Second International Conference on Rough Sets and Current Trends in Computing RSCTC'2000*, pages 340–347, 2000.
- [7] R. Kohavi, D. Sommerfield, and J. Dougherty. Data Mining using MLC++: A Machine Learning Library in C++. *Tools with Artificial Intelligence*, pages 234–245, 1996.
- [8] K. Lakshminarayan, S. A. Harp, and T. Samad. Imputation of Missing Data in Industrial Databases. *Applied Intelligence*, 11:259–275, 1999.
- [9] Hwei Diana Lee, Maria Carolina Monard, and José Augusto Baranauskas. Empirical Comparison of Wrapper and Filter Approaches for Feature Subset Selection. Technical Report 94, ICMC - USP, São Carlos, SP, Oct 1999. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel.tec/Rt_94.ps.zip.
- [10] R. J. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. John Wiley and Sons, New York, 1987.
- [11] C. J. Merz and P. M. Murphy. UCI Repository of Machine Learning Datasets, 1998. <http://www.ics.uci.edu/mlearn/MLRepository.html>.
- [12] J. R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann, CA, 1988.
- [13] D. R. Wilson and T. R. Martinez. Reduction Techniques for Exemplar-Based Learning Algorithms. *Machine Learning*, 38(3):257–286, March 2000.

Determining The Degree of Generalization Using An Incremental Learning Algorithm

Pablo Zegers

*Facultad de Ingeniería, Universidad de los Andes
San Carlos de Apoquindo 2200, Las Condes, Santiago, Chile
pzegers@uandes.cl*

Malur K. Sundareshan

*Electrical and Computer Engineering Department, The University of Arizona
1230 East Speedway, Tucson, AZ 85721-0104, USA
sundareshan@ece.arizona.edu*

Abstract. Any Learning Machine (LM) trained with examples poses the same problem: how to determine whether the LM has achieved an acceptable level of generalization or not. This work presents a training method that uses the data set in an incremental manner such that it is possible to determine when the behavior displayed by the LM during the learning stage truthfully represents its future behavior when confronted by unseen data samples. The method uses the set of samples in an efficient way, which allows discarding all those samples not really needed for the training process. The new training procedure, which will be called "Incremental Training Algorithm", is based on a theoretical result that is proven using recent developments in statistical learning theory. A key aspect of this analysis involves identification of three distinct stages through which the learning process normally proceeds, which in turn can be translated into a systematic procedure for determining the generalization level achieved during training. It must be emphasized that the presented algorithm is general and independent of the architecture of the LM and the specific training algorithm used. Hence it is applicable to a broad class of supervised learning problems and not restricted to the example presented in this work.

1 Introduction

This paper focuses on an incremental learning algorithm devised to determine whether a trained Learning Machine (LM) has reached an acceptable level of generalization or not using the set of samples in an efficient manner, just from an examination of the learning behavior of the LM.

The issue about generalization has been a main concern since learning problems started to be studied. Thus, it is not a surprise that the problem was addressed when the learning automaton concept [12, 17, 6, 11] was born in the early sixties. Because the question about generalization bears direct impact on the adaptation process, it constitutes a core question of the learning automata problem. Nevertheless, due to the stochastic nature of these machines it has proven difficult to understand the generalization process in the context of the learning automaton, and, therefore, difficult to optimally tailor the training process of these machines in order to improve their generalization capability. Thus, in terms of the analysis of its generalization capacity, the learning automaton remains largely a heuristic method. During the last

20 years, the research community has studied different aspects of the generalization problem in a broader class of LMs [1, 18, 19, 2, 4, 3]. While all these works have attempted to tackle different aspects of the problem, development of systematic procedures for determining when a generic LM has attained an acceptable level of generalization has proven very difficult to attain. It is important to keep in mind that the goal in any training procedure is to obtain a reasonable level of generalization using a minimum of samples and computational resources.

An important exception to this is the recent understanding gained on the support vector machine [14, 15, 16, 8]. The usefulness of the support vector machine method has been proven by a cadre of practical algorithms and theoretical results that allow one to ensure when a desired level of generalization has been reached and to determine which samples are essential for the training process, *a.k.a.* to find the support vectors. Even though this architecture works as a universal function approximator, and therefore it is the basis for a very important and useful class of LMs, these results only apply for this specific architecture and they have not been generalized for arbitrary architectures.

In Summary, there is still no theory or organized methods that provide practical answers to how to determine the degree of generalization with an efficient usage of the set of samples in the case of a generic LM. This work presents an incremental learning method that under certain sufficient conditions permits to answer the question of when a good degree of generalization has been attained while keeping the usage of samples to a minimum. The main advantage of this method is that it only requires studying some key aspects underlying the learning behavior of the LM. The paper starts with an introduction to statistical learning theory [14, 15, 16], continues presenting the new learning scheme, analytically proves why the presented algorithm should work, tests everything with an experiment, and ends with a discussion of the results.

2 Statistical Learning Theory

The LM problem, as defined in this work, consists in finding $\theta_* = \arg \min_{\theta} E_I(\theta)$, where

$$E_I(\theta) = \int dx f_X(x) (y - \hat{y})^2 \quad (1)$$

is called the “intrinsic error”, $f_X(x)$ is a density function, $y = g(x)$ is a reference function, and $\hat{y} = \hat{g}(x, \theta)$ is an estimation of the reference function defined by the parameter vector θ . However, what makes the problem complicated is that $f_X(x)$ is unknown and, therefore, it is not possible to find θ_* . Instead, a set of l independent and identically distributed data samples $y_i = g(x_i)$ is available, such that it is only possible to find $\theta_l = \arg \min_{\theta} E_E(\theta, l)$, where

$$E_E(\theta, l) = \frac{1}{l} \sum_{i=0}^l (y_i - \hat{y}_i)^2 \quad (2)$$

is the so called “empirical error”. The reader is referred to the recent texts [14, 15, 16], which provide an excellent treatise on the basic concepts of statistical learning theory.

Given the previous formalism, it is clear that in order to measure the degree of generalization, *i.e.* measure the intrinsic error, it is necessary to ensure that the learning process will produce a sequence of θ_l as l is increased that converges to θ_* , *i.e.* it is necessary to ensure the consistency of the procedure. If this is not achieved, there is no guarantee that the learning procedure will eventually produce θ_* . Given that the consistency stage is reached, then the learning procedure associated with the LM will necessarily produce a corresponding

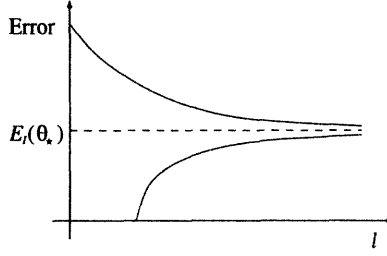


Figure 1: Intrinsic and empirical error vs. number of samples. The upper curve represents the $E_I(\theta_l)$ and the lower curve $E_E(\theta_l, l)$, both with $\theta_l = \arg \min_{\theta} E_E(\theta, l)$

empirical error that tends to the intrinsic error, and thus it becomes possible to measure the intrinsic error.

Given a LM, Fig. 1 depicts a typical evolution of these two error measures, *i.e.* intrinsic and empirical errors, evaluated at $\theta = \theta_l = \arg \min_{\theta} E_E(\theta, l)$ as the set of samples grows [14, 15, 16]. It can be seen that $E_E(\theta_l, l)$ is always less than $E_I(\theta_l)$. The reason is that while θ_l minimizes $E_E(\theta, l)$, which takes into account a set of l data samples (thus providing l constraints for establishing a match), θ_l does not minimize $E_I(\theta)$, which considers the same l data samples as well as all other possible ones. Only as $l \rightarrow \infty$ is that the both error measures converge to the same value when evaluated for $\theta = \theta_l$.

Fig. 1 shows that the behavior of $E_E(\theta_l, l)$ can be divided into three different stages: an initial one that corresponds to a low number of data points in the set of samples, during which the LM memorizes each of the events such that $E_E(\theta_l, l)$ can be made arbitrarily close to zero (the LM shatters the problem space), a second stage, associated to an increased number of samples, where the LM no longer can memorize the samples and $E_E(\theta_l, l)$ starts to grow and move towards $E_I(\theta_l)$ (the data samples start to drive the LM towards the desired solution), and finally the third stage, where $E_E(\theta_l, l)$ is close to $E_I(\theta_l)$ and the learning process is finally consistent (minimization of the intrinsic error is now possible).

One of the main results of statistical learning theory [13, 14, 15, 16] states that when $A \leq E_I(\theta)$, $E_E(\theta, l) \leq B$, then for any positive ϵ

$$P \left\{ \sup_{\theta} |E_I(\theta) - E_E(\theta, l)| > \epsilon \right\} \leq \min \left\{ 1, 4 \exp \left(\left(\frac{h(\ln \frac{2}{\epsilon} + 1)}{l} - \frac{(\epsilon - \frac{1}{B-A})^2}{(B-A)^2} \right) l \right) \right\} \quad (3)$$

which bounds the probability of worst, *i.e.* supremum over θ , divergence between $E_I(\theta)$ and $E_E(\theta, l)$ [14, 15, 16]. The constant h is called the VC dimension of the LM. The exponent of the exponential function on the right side of Eq. 3 has two terms. The first one grows logarithmically in l , and the second one decreases linearly in l . What is important to notice is that if $h < \infty$, the second term takes over as l grows and drives the exponential to zero.

The condition expressed in Eq. 3 confirms that the convergence process is divided into the three different stages mentioned before: a first one where the probability of divergence keeps close to one (memorization stage), a second one that signals the onset of the exponential effect (transition stage), and a final stage where the probability of divergence between the two errors is close to 0 and the LM is in a position where it can generalize (consistency stage).

It is important to stress that this result states that a LM with a finite VC dimension will achieve its optimal configuration in the sense that it will converge towards producing the minimum intrinsic error $E_I(\theta)$ as the number of samples grows, but it does not guarantee that the intrinsic error will be zero. In other words, this result provides conditions that establish

under which conditions a LM will do its best, but it does not ensure that the LM will do the best, *i.e.* achieve an intrinsic error equal to zero.

In general terms, finiteness of the VC-dimension provides the necessary and sufficient conditions for distribution independent consistency, *i.e.* convergence of the empirical error to the intrinsic error, which is a necessary step in order to allow the LM to find a combination of parameters that minimizes the intrinsic error. This result is the cornerstone of statistical learning theory and its strength rests on being a distribution independent result that guarantees exponential convergence even in a worst case scenario.

3 The Incremental Learning Algorithm

Consider a student that has to learn a certain topic. If this is an efficient student, he certainly does not go to the library, check out all the books that contain material related to the topic, and study all of them. Instead, the student first gets a primer on the subject, reads it, and solves the problems contained in it. Then, he checks out another book, ignores all the material he has already mastered and focuses on learning whatever new material there is in the new book. He continues doing so until he cannot find new information or problems he does not know how to handle. It is then, and only then, that the student can be sure that he has mastered the topic. Two things that can be observed about the learning process used by the student are:

1. The student focuses on learning only that information that brings up new aspects of the problem to his knowledge, and ignores the rest.
2. A telltale sign that the student is becoming closer to mastering the topic is the increasing difficulty in finding new material.

A learning method that is inspired by the example cited above is described in the flow diagram shown in Fig. 2. This algorithm will be termed “incremental learning algorithm” in future discussion.

In the incremental learning algorithm, the term “training event” refers to the execution of all the steps in the algorithm from the moment the empirical error $E_E(\theta, l) > \delta$ and the algorithm branches towards the “Train LM” step, until the empirical error complies with $E_E(\theta, l) < \delta$ and the algorithm branches towards the “Set $l = l + 1 \dots$ ” step. In this incremental learning algorithm, the threshold δ represents the bound on the intrinsic error that is deemed acceptable to the user.

One may note some similarities between the incremental learning algorithm and the classic perceptron learning rule [5]. However, there is an important difference between them: in the perceptron learning rule each training event only uses the sample that made the system fail. In the perceptron learning rule there is no notion of a training set at all and training is done only with the last sample that was processed by the system. On the other hand, the incremental learning rule presented above uses a training set composed of all the samples previously processed by the system. The analysis that follows proves that this seemingly minor difference is very important.

The following theorem proves that under certain conditions the incremental learning algorithm behaves like the efficient student in the motivating example discussed above:

Theorem 1. *In every LM trained with the incremental learning algorithm, right after the training event that started when the l^{th} sample was examined ends*

$$P \{ E_I(\theta) > \delta \} \leq \min \left\{ 1, 4 \exp \left(\left(\frac{h(\ln \frac{l}{l} + 1)}{l} - \frac{(\delta - E_E(\theta, l) - \frac{\delta}{2})^2}{(B - A)^2} \right) l \right) \right\} \quad (4)$$

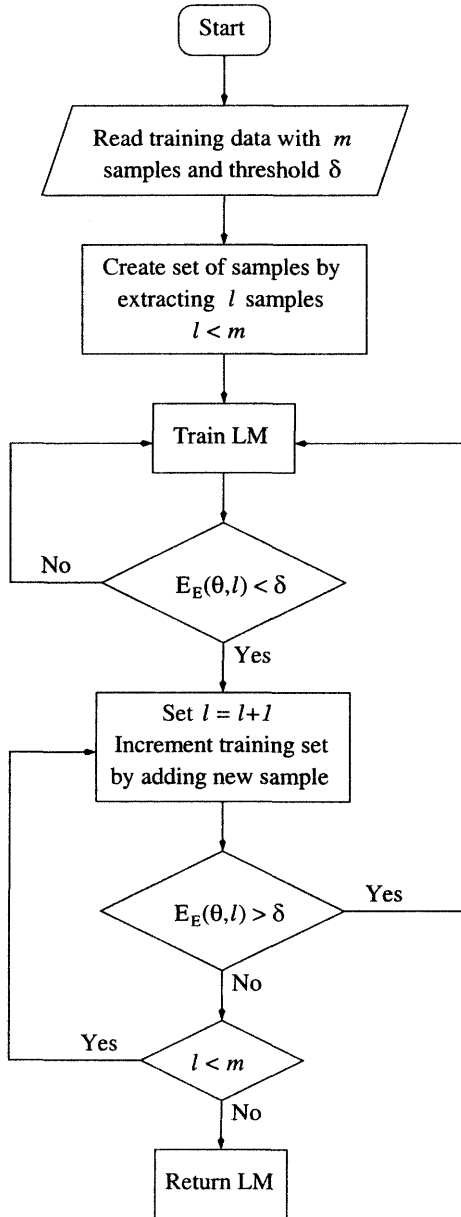


Figure 2: Flow diagram of the incremental learning algorithm.

where θ is the vector of parameters obtained at the end of the training event.

Proof: As explained before (see Eq. 3), when $A \leq E_I(\theta)$, $E_E(\theta, l) \leq B$, then for any positive ϵ

$$P \left\{ \sup_{\theta} |E_I(\theta) - E_E(\theta, l)| > \epsilon \right\} \leq \min \left\{ 1, 4 \exp \left(\left(\frac{h(\ln \frac{2l}{l} + 1)}{l} - \frac{(\epsilon - \frac{1}{l})^2}{(B-A)^2} \right) l \right) \right\} \quad (5)$$

If the previous expression is evaluated for an arbitrary θ , then

$$P \{ |E_I(\theta) - E_E(\theta, l)| > \epsilon \} \leq \min \left\{ 1, 4 \exp \left(\left(\frac{h(\ln \frac{2l}{l} + 1)}{l} - \frac{(\epsilon - \frac{1}{l})^2}{(B-A)^2} \right) l \right) \right\} \quad (6)$$

Then, it is also true that

$$P \{ E_I(\theta) - E_E(\theta, l) > \epsilon \} \leq \min \left\{ 1, 4 \exp \left(\left(\frac{h(\ln \frac{2l}{l} + 1)}{l} - \frac{(\epsilon - \frac{1}{l})^2}{(B-A)^2} \right) l \right) \right\} \quad (7)$$

Rearranging the terms in the left side

$$P \{ E_I(\theta) > \epsilon + E_E(\theta, l) \} \leq \min \left\{ 1, 4 \exp \left(\left(\frac{h(\ln \frac{2l}{l} + 1)}{l} - \frac{(\epsilon - \frac{1}{l})^2}{(B-A)^2} \right) l \right) \right\} \quad (8)$$

In the incremental learning algorithm, right after a training event finishes $\delta > E_E(\theta, l)$. Therefore, it is possible to define $\epsilon = \delta - E_E(\theta, l) \geq 0$. Replacing into the previous expression

$$P \{ E_I(\theta) > \delta \} \leq \min \left\{ 1, 4 \exp \left(\left(\frac{h(\ln \frac{2l}{l} + 1)}{l} - \frac{(\delta - E_E(\theta, l) - \frac{1}{l})^2}{(B-A)^2} \right) l \right) \right\} \quad (9)$$

□

This theorem proves that if a LM with $h < \infty$ is trained with the incremental learning algorithm, right after a training event finishes the probability that the intrinsic error is above the threshold δ converges to zero exponentially after a certain number of samples has been processed. Because the empirical error is bounded by a shrinking confidence interval centered on the intrinsic error (see Eq. 3), the exponential behavior ends up influencing the empirical error too. Therefore, the probability of a training event getting triggered will exhibit the three different stages mentioned before:

1. A first one that corresponds to a low number of samples where the probability of divergence between the empirical error $E_E(\theta, l)$ and the intrinsic error $E_I(\theta, l)$ is one, and the probability that $E_E(\theta, l) > \delta$ is also close to 1. Therefore, the probability that a training event gets triggered is close to 1.
2. A second one, associated to an increased number of samples, where the probability that $E_E(\theta, l) > \delta$ starts to decrease, lowering the probability that a training event gets triggered.
3. A last stage where the learning process has become consistent and the empirical error behaves like the intrinsic error. It is during this stage that the probability that $E_E(\theta, l) < \delta$ converges to 1 and no more training events are triggered.

The importance of the previous result rests on the fact that if no more training events are triggered, *i.e.* the empirical error keeps below δ , it is possible to say that the learning process has become consistent and that it has effectively found a parameter combination that produces an intrinsic error less than δ . Once this can be assessed it is possible to stop the training process while having a high certainty that the intrinsic error of the LM will keep below δ for the future samples that the LM may encounter within the data set, *i.e.* the LM has achieved a desired level of generalization. Once this point is reached, because the LM has found the parameters that produce an intrinsic error below δ , there is no need to continue testing and looking for examples that make the system fail and the rest of the set of samples can be discarded.

4 Experiments

In this section the results of an experiment that employed a neural network as the architecture for the LM will be described. It must be emphasized that the architecture selection was only for illustrative purposes and the present algorithm is applicable to any other architecture that may be selected for configuring the LM. A multilayer perceptron with 2 input neurons, 1 hidden layer with 50 neurons, another hidden layer with 25 neurons, and 1 output neuron was selected as LM, and used to learn a 2D sinc $z = \sin 6\pi \sqrt{x^2 + y^2} / (12\pi \sqrt{x^2 + y^2})$. Initialization of this neural network was performed using a scheme recommended in [7], and the network was trained with the RPROP algorithm [9, 10], which is one of the algorithms that is currently implemented in the Matlab Neural Networks Package. Since our interest in this work is to demonstrate the ability of the present incremental learning algorithm in testing the generalization level, the specific training procedure selected for the neural network architecture is only illustrative. The set of samples used for training was obtained by randomly sampling within the interval $[-0.5, 0.5]$ according to a uniform density function in order to obtain x and y , and generating the corresponding outputs z using the 2D sinc function. In order to obtain an estimate of the average behavior of the LM, 20 different sets of initial conditions and 20 different sets of samples were employed. The LM was trained using 20 different combinations of sets of initial conditions and sets of samples, making sure that no set of initial conditions and no set of samples were used more than once. Training was done using the incremental learning algorithm presented above by setting an intrinsic error threshold $\delta = 10^{-4}$ (set arbitrarily) and $l_0 = 100$ (determined experimentally). The reference 2D sinc used in the training process is shown in Fig. 3. A 2D sinc generated after one of the training runs is shown in Fig. 4.

The number of samples examined by the incremental learning algorithm between individual training events is shown Fig. 5. The estimated probability of a training event getting triggered is shown in Fig. 6. This probability is calculated by dividing the number of times that the k^{th} training event happened by the number of different sets of initial conditions (which in this experiment is 20). Both plots clearly show that less than 300 training events sufficed to make all 20 training runs reach the desired consistency stage.

5 Discussion

The experiment of learning the 2D sinc function outlined above demonstrated that the probability of a training event getting triggered exhibits three stages as expected: a first one where training events follow continuously, one after the other, a transition zone where the probability of triggering a training event starts to decrease, and the last one, where the probability that the empirical error exceeds the defined threshold converges to zero.

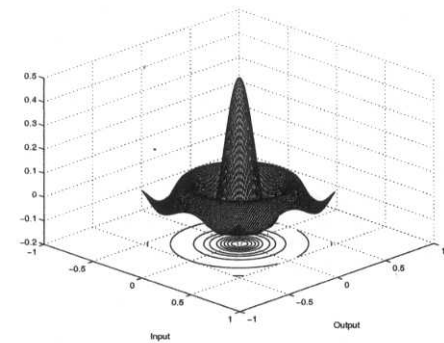


Figure 3: Reference 2D sinc used in the training process.

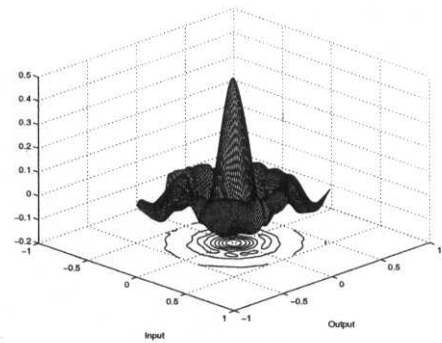


Figure 4: Output generated by the LM after training (2D sinc).

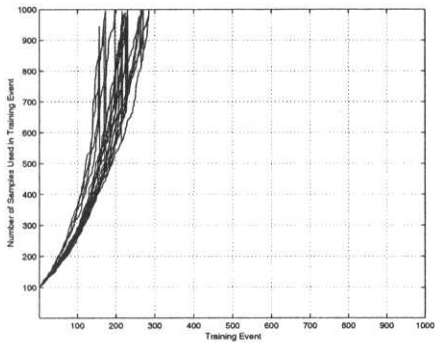


Figure 5: Number of samples skipped as a function of the number of training events (2D sinc).

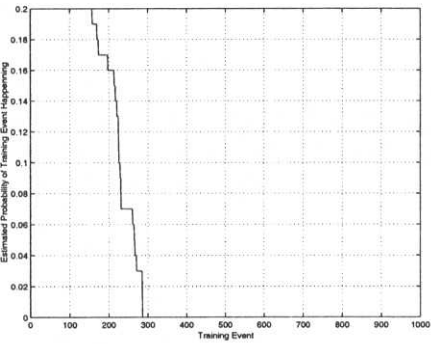


Figure 6: Estimated probability of triggering a training event as a function of the training event number (2D sinc).

While the above experiment illustrates a successful application of the present algorithm, several other experiments that we have conducted indicate that under some conditions the incremental learning method previously described may not show signs of reaching the consistency stage, *i.e.* the third stage in the evolution of the learning process. There are two reasons that explain this behavior:

1. The LM keeps memorizing the data, the number of elements in the set of samples increases constantly, and the probability that a training event happens keeps high. This happens when there are not enough data samples or the VC dimension of the LM is not finite.
2. The minimum value of the intrinsic error can not comply with $E_I(\theta) < \delta$. In this case the LM starts a training event but never finishes it. This can happen if the δ is too low or if the initial conditions, learning procedure, or stopping conditions impede the LM from reaching empirical error values below those it should reach. This happens when the LM falls into a local minima and it is not able to escape out of it.

While the first cause is relatively simple to remedy (it suffices to use more samples or a different architecture), the second is more complex because it is associated to problems not well resolved for arbitrary LMs. Even though higher values of δ could be used, there is no sure way of avoiding the effects of the starting conditions, the learning procedure, or the stopping rule when it comes to getting stuck at a local minimum.

6 Conclusions

The method presented in this paper provides with a practical way of determining whether a training process has attained the consistency stage or not. As a consequence of this, it is possible to determine if the training process, which is focused on minimizing the empirical error, is really minimizing the intrinsic error or not. Therefore it is possible to establish whether an LM has attained an acceptable degree of generalization or not. Once this is determined to be true it is possible to stop the training process and safely ignore all the other data points in the set of samples.

Also, thanks to the theorem proven in this work it is possible to say that the behavior seen in the experiments should also be observed in any LM. Therefore, the present results hold for any architecture and make the incremental learning algorithm a valid approach for determining the degree of generalization in a general setting using the set of samples in an efficient manner.

A particularly desirable aspect of this method is that whether the LM has reached a good level of generalization or not can be deduced from its learning behavior and there is no need for using analytical methods or probe sets in order to test this. There is no need for finding the VC dimension of the LM in order to check if it has generalized or determining the number of samples needed to do so: if the probability that a training event gets triggered goes to zero, then the system has already reached the consistency stage and the empirical error is a good measure of its level of generalization.

7 Acknowledgements

Special thanks to Dr. Mark Neifeld for his comments on how to improve this work.

References

- [1] Baum, E.B., Neural Net Algorithms That Learn in Polynomial Time from Examples and Queries, *IEEE Transaction on Neural Networks* 2(1) (1991) 5–19.
- [2] Cachin, C., Pedagogical Pattern Selection Strategies, *Neural Networks* 7(1) (1994) 175–181.
- [3] Cataltepe, Z. and Abu-Mostafa, Y.S. and Magdon-Ismael, M., No Free Lunch for Early Stopping, *Neural Computation* 11 (1999) 995–1009.
- [4] Franco, L. and Cannas, S.A., Generalization and Selection of Samples in Feedforward Neural Networks, *Neural Computation* 12 (2000) 2405–2426.
- [5] Haykin, S.S., *Neural Networks: A Comprehensive Foundation*, Prentice Hall (1998).
- [6] Narendra, K. and Thathachar, M.A.L., *Learning Automata: An Introduction*, Prentice Hall (1989).
- [7] Nguyen, D. and Widrow, B., Improving the Learning Speed of 2-Layer Neural Networks by Choosing Initial Values of The Adaptive Weights, *Proceedings of the IJCNN* (1990).
- [8] Ralaivola, L. and d'Alché-Buc, F., Incremental Support Vector Machine Learning: A Local Approach, *Proceedings of the ICANN*, Vienna, Austria (2001).
- [9] Riedmiller, M. and Braun, H., A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm, *Proceedings IEEE International Conference on Neural Networks*, San Francisco, USA (1993).
- [10] Riedmiller, M., RPROP – Description and Implementation Details, *University of Karlsruhe* (1994).
- [11] Sundareshan, M.K. and Condarcure, T.A., Recurrent Neural-Network Training by a Learning Automaton Approach for Trajectory Learning and Control System Design, *IEEE Transactions on Neural Networks* 9(3) (1998) 354–368.
- [12] Tsetlin, M., On The Behavior of Finite Automata in Random Media, *Automation and Remote Control* 22 (1961) 1210–1219.
- [13] Vapnik, V. and Levin, E. and Le Cun, Y., Measuring The VC-Dimension of a Learning Machine, *Neural Computation*, 6 (1994) 851–876.
- [14] Vapnik, V.N., *The Nature of Statistical Learning Theory*, Springer (1995).
- [15] Vapnik, V.N., *Statistical Learning Theory*, John Wiley and Sons (1998).
- [16] Vapnik, V.N., An Overview of Statistical Learning Theory, *IEEE Transactions on Neural Networks* 10(5) (1999) 988–999.
- [17] Varshavskii, V.I. and Vorontsova, I.P., On The Behavior of Stochastic Automata with Variable Structure, *Automation and Remote Control* 24 (1963) 327–333.
- [18] Zegers, P., Reconocimiento de Voz Utilizando Redes Neuronales, *Engineer Thesis*, Pontificia Universidad Católica de Chile, Chile (1992).
- [19] Zegers, P. and Sundareshan, M.K., Optimal Tailoring of Trajectories, Growing Training Sets, and Recurrent Networks for Spoken Word Recognition, *Proceedings of the ICNN*, Anchorage, USA (1998).

Towards Landscape Analyses to Inform the Design of a Hybrid Local Search for the Multiobjective Quadratic Assignment Problem

Joshua D. Knowles¹ and David W. Corne²

¹ Free University of Brussels

² University of Reading

Abstract. The quadratic assignment problem (QAP) is a very difficult and practically relevant combinatorial optimization problem which has attracted much research effort. Local search (LS) moves can be quickly evaluated on the QAP, and hence favoured methods tend to be hybrids of global optimization schemes and LS. Here we introduce the *multiobjective* QAP (mQAP) where $m \geq 2$ distinct QAPs must be minimized simultaneously over the same permutation space, and hence we require a set of solutions approximating the Pareto front (PF). We argue that the best way to organise a hybrid LS for the mQAP will depend on details of the multiobjective fitness landscape. By using various techniques and measures to probe the landscapes of mQAPs, we attempt to find evidence for the relative ease with which the following can be done by LS: approach the PF from a random initial solution, or search along or close to the PF itself. On the basis of such explorations, we hope to design an appropriate hybrid LS for this problem. The paper contributes a number of landscape measurement methods that we believe are generally appropriate for multiobjective combinatorial optimization.

1 Introduction

Many problems can be formulated as a *quadratic assignment problem* (QAP). Computationally, this is NP hard [10] and even relatively small general instances ($n \geq 20$) cannot be solved to optimality. We introduce here the *multiobjective* QAP (mQAP) as both a practically important problem and a benchmark that we believe will be useful in general research on multiobjective combinatorial landscapes. As in the scalar QAP, fast local search (LS) is possible because a difference in the objective function value(s) can be calculated quickly when using a 2-exchange operator.

The question of interest is how to control and organise the many required runs of LS. Depending on the landscape, it may not be efficient to start each search *ab initio*. Rather, it may be better to search in parallel for the different Pareto optima, by alternating the ‘search direction’ in the objective space, step by step. Or, alternatively again, the best approach may be to first find one Pareto optimum and then to search along the Pareto front from this one to find others.

Our aim here is to propose some multiobjective landscape analysis methods that might be used to predict the best approach given an unseen mQAP instance. The paper is necessarily

speculative as little work has been done on the best way to apply hybrid LS in multiobjective domains and still less on multiobjective landscape analysis.

The remainder is organized as follows. Section 2 reviews the scalar QAP and describes a best improvement LS that can also be used for the mQAP. Section 3 introduces the mQAP, proposing applications and its theoretical interest. A test instance generator is described in Section 4. Section 5 outlines different overall search strategies for approximating the PF of the mQAP, relating these to landscape correlations. Tools for measuring properties of the multiobjective landscape are described in Section 6, and some illustrative results are given. Section 7 concludes.

2 The scalar QAP

The quadratic assignment problem (QAP) is the problem of assigning each of n facilities to exactly one of n possible locations so as to minimize the sum of products of flows between the facilities and the distances between the locations. For a mathematical formulation of the problem see that for the mQAP in section 3, the only difference being the use of multiple ($m \geq 2$) flow matrices. Solutions to the QAP may be represented by a permutation π on the set $\{1, \dots, n\}$, where n is the number of facilities/locations.

A deterministic best-improvement local search for the QAP, based on exchanging the locations of two facilities, was proposed by Taillard [3]. The search is fast because it is possible to calculate the change in cost of an exchange in $O(n)$ time as only the two affected rows and columns in the flow matrix must be re-evaluated. Further, it is possible to use solutions from previous iterations to calculate some of the exchanges in just $O(1)$ time.

Much work has been concerned with measures on the fitness landscape of QAP instances. Here we will just give those used later in this paper, which mainly depend on a measure of distance in the parameter (permutation) space. For example, Bachelet [1] measures distance $\text{dist}(\pi, \mu)$ between two solutions π and μ as the smallest number of 2-swaps that must be performed to transform one solution into the other (this can be computed in $O(n)$ time); this distance measure has a range of $[0, n - 1]$. From this, other measures can be easily defined. Bachelet gives the diameter of a population P as

$$\text{dmm}(P) = \frac{\sum_{\pi \in P} \sum_{\mu \in P} \text{dist}(\pi, \mu)}{|P|^2}.$$

The entropy [4], which is a further measure of the dispersion of solutions, is given by

$$\text{ent}(P) = \frac{-1}{n \log n} \sum_{i=1}^n \sum_{j=1}^n \left(\frac{n_{ij}}{|P|} \log \frac{n_{ij}}{|P|} \right)$$

where n_{ij} is the number of times facility i is assigned to location j in the population. The fitness distance correlation [9] has been widely used. Often a plot is shown, either giving the fitness against distance from the nearest global optimum or best known solution, or all pairs of solutions may also be differentiated.

Vollmann and Buffa [12] introduced flow dominance as a means of characterizing QAP instances. Flow dominance is a measure of the flow matrix given by

$$fd(c) = 100 \times \frac{a}{b} \quad a = \sqrt{\frac{\sum_{i=1}^n \sum_{j=1}^n (c_{ij} - b)^2}{n^2}} \quad b = \frac{\sum_{i=1}^n \sum_{j=1}^n c_{ij}}{n^2}$$

When there is high flow dominance there is low epistasis, in general. The distance dominance dd can be defined on the distance matrix in an analogous fashion.

3 The multiobjective QAP model

The multiobjective QAP (mQAP), with multiple flow matrices (defined below) naturally models any facility layout problem where we are concerned with the flow of more than one type of item or agent. For example, in a hospital layout problem we may be concerned with simultaneously minimizing the flows of doctors on their rounds, of patients, of hospital visitors, and of pharmaceuticals and other products. We may imagine that each of the matrices describing these flows is very different and that until we see the Pareto front it may not be possible to simply weight the importance of each flow. Thus we have a very natural multi-objective problem. Similar examples can be given for other facility location problems such as the layout of factory floors and the topology of distribution networks. Furthermore, other problems that can be modeled using QAP, such as scheduling, may also exist in multiobjective form. The mQAP is also likely to be a useful problem to study from a theoretical viewpoint.

Formally, the mQAP problem is to ‘minimize’:

$$\vec{c}(\pi) = \{c^1(\pi), c^2(\pi), \dots, c^m(\pi)\}, \text{ where } c^k(\pi) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} b_{\pi_i \pi_j}^k, \quad k \in 1..m$$

where n is the number of facilities/locations, m is the number of different flow matrices, a_{ij} is the distance between location i and location j , b_{ij}^k is the k th flow from facility i to facility j , and π_i gives the location of facility i in permutation $\pi \in P(n)$, where $P(n)$ is the set of all permutations of $\{1, 2, \dots, n\}$.

The term ‘minimize’ must also be defined because different methods are available for tackling multiobjective problems, including lexicographic ordering, scalarizing techniques, or constraint methods. However, it is often undesirable to use any of the aforementioned techniques, which induce a total ordering on the solutions, without knowledge of what trade-offs of the objectives are ‘out there’ to be found. In this case it is best to use Pareto optimization, a technique that makes the fewest possible assumptions about which solutions are preferable, before the search is conducted. Here, the aim of Pareto optimization is to find a set of solutions $P^*(n) \subseteq P(n)$, the Pareto optimal set, which are *nondominated* in the whole search space:

$$\begin{aligned} &\forall \pi^* \in P^*(n) \nexists \pi \in P(n) \text{ such that } \pi \prec \pi^*, \text{ where } \pi \prec \pi^* \text{ iff} \\ &\forall k \in \{1, \dots, m\} \quad c^k(\pi) \leq c^k(\pi^*) \wedge \exists k \in \{1, \dots, m\} \text{ such that } c^k(\pi) < c^k(\pi^*) \end{aligned}$$

where $\pi \prec \pi^*$ is read as π *dominates* π^* . The set of objective vectors (or points) $C^* \subseteq C$, called the Pareto front (PF), is the image of the Pareto optimal set in the objective space, $\vec{c}(P^*(n)) \mapsto C^*$, and C is the image of the whole permutation space $\vec{c}(P(n)) \mapsto C$. The aim of Pareto optimization is to find a good approximation to the whole Pareto front C^* . To understand what is meant by a good approximation, the reader is referred to [7] but need only know that sets of solutions (approximation sets) can be partially ordered without additional assumptions.

4 An mQAP instance generator

To obtain instances for the mQAP we have developed a generator that allows different problem features to be controlled. The generator makes symmetric multi-objective QAP instances

with one distance matrix and multiple flow matrices. The distance and flow matrices have structured, 'real-world-like' entries [3] and correlations can be set between corresponding entries in the two or more flow matrices. The generator follows procedures for making the non-uniformly random QAP problems given the appellation TaiXXb in the literature, outlined in [3].

The number of flow matrices is controlled by a parameter, m . With $m \geq 2$, a multi-objective QAP problem is generated with m flow matrices. The entries in the k th matrix ($2 \leq k \leq m$) are generated where a random variable X is correlated with the value of X used in the corresponding entry in the first flow matrix. Correlations between -1 and 1 can be set between the first and each of the additional flow matrices using $m - 1$ further parameters to the generator.

A degree of overlap between the matrices can also be specified. The overlap parameter is set between 0 and 1 . It controls the fraction of entries in the j th flow matrix that are correlated with the corresponding entries in the 1st flow matrix. With the overlap parameter set to zero, a random un-correlated value will be placed in each entry of the j th flow matrix that corresponds to a zero entry in the first flow matrix. Similarly, a zero will be placed in each entry of the j th flow matrix that corresponds to a non-zero value in the first flow matrix. Thus there is no overlap between the flows of the first and j th matrix. With the overlap parameter set to 1 all the flows are correlated. With the overlap set to intermediate values some of the flows will overlap and others will not.

5 Hybrid search schemes for the mQAP

Multiobjective (MO) metaheuristics for Pareto optimization have been around for quite some time now and MO versions of genetic algorithms [5], simulated annealing [2], tabu search (TS) [6], and others have been proposed. Some hybrid genetic algorithm approaches have also been put forward, e.g. [8]. With the mQAP, the availability of a fast best-improvement LS (or TS) means that any hybrid approach should consider employing LS by specifying a scalarizing weight vector and allowing the LS to find a local optimum of this search 'direction'. The question is, how should the runs of the LS be controlled by the overall search strategy? This question is already difficult in scalar optimization. We believe the answer concerns the best way to move about in the multiobjective landscape.

Figure 1 illustrates three quite different strategies, using a 2-objective problem as an example. The first strategy is to use a population-based approach which tries to gradually improve the population in all directions at once, approaching the PF from all angles, with LS applied to all solutions in the population at each generation. In the second strategy, a scalarizing weight vector is first chosen and the LS is applied multiple times in only this direction until no more improvement results. Each LS re-start could be from a random point but more usually it would be from a perturbation of the previous local optimum, taking it out of that basin of attraction [11]. After one point on or near the PF is found the process is repeated for a different scalarizing vector. In the last strategy we envisage moving towards the PF in one direction by repeated LS applications as in the previous method. But then once on the PF, the same solution should be perturbed again and minimized over a 'nearby' scalarizing vector, resulting in a point nearby on the PF.

Clearly, the best approach will depend on whether or not points nearby in objective space are also nearby in the permutation space. In particular, we can consider this in two different

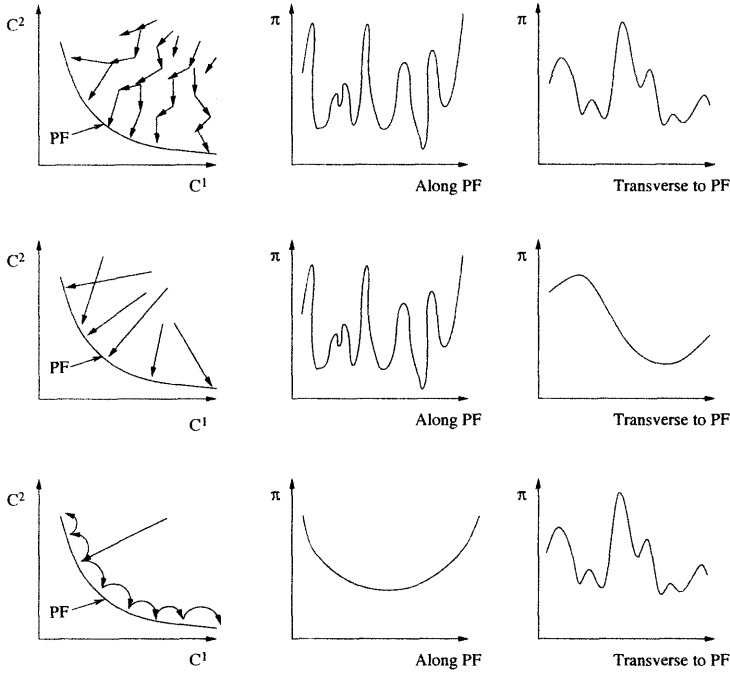


Figure 1: Ways to approach the Pareto front and the underlying landscape appropriate to each. The left column shows the Pareto front of a two-objective problem and different ways a search algorithm may find points close to the PF. The centre and right columns depict correlations between the multiobjective fitness and the distance between points in the parameter space. See main text for further explanation

ways. For a particular scalarizing vector is it easy to move towards (perpendicular to) the PF i.e. is the fitness distance correlation good in this search direction? And for a set of mutually nondominated points (i.e. for points sharing a common ‘fitness’ but not in the same objectives), is it easy to move from one point to others nearby, parallel to the local Pareto front? It is possible that the degree of correlation in the landscape varies tremendously depending on which direction of search is considered and/or what is the fitness level. It may be very difficult to find meaningful measurements of these properties and even more difficult to predict which will be the best method of search. Nonetheless, it seems likely that the success of searches will depend heavily on what kind of overall strategy is employed, and this in turn on the landscape, so it seems well worth trying to gain some understanding of local correlations as well as global properties. In the next section we describe some example measurements we have made and discuss what they could mean.

6 Landscape measurement tools and example results

Our methods are based on those previously proposed for the scalar QAP, as briefly reviewed in section 2.1 (and thoroughly described in [1]). The overall strategy in the scalar QAP case is to run a local search from several random starting solutions. Measuring different properties

Table 1: Results of our measurements on two 20 node, 2 flow matrix, and two 30 node, 3 flow matrix instances with different flow correlations and other parameters. The first figure in an instance name is the number of nodes, the second is the number of flow matrices and the third is just an index. The global properties are: $\max(d)$, the maximum distance in the distance matrix; $\max(f)$ the maximum flow in any of the flow matrices; $\text{corr}(f^i, f^j)$ the correlation between corresponding flow matrix entries of the i th and j th flow; the overlap of correlated entries; fd^k , the flow dominance of the k th flow matrix; and dd the distance dominance. The local search measures were collected from 100000 (105000, 3 objective) starting points. Figures in brackets are sample sizes. Where U or O is primed this means that the measure is calculated on a sample of this set. The sampling criteria are as follows: rs refers to a random sample; $\bar{\lambda}$ to solutions satisfying the condition, $\sum_{j=1}^k (\lambda^j - \xi^j)^2 < 0.01$ where $\xi = (0.0, 0.3, 0.7)$ for the three objective instances, and $\xi = (0.1, 0.9)$ for the two-objective instances; and nds refers to the nondominated solutions from the full set

Glob. Property	Instances			
	Kno20-2fl-1rl	Kno20-2fl-2rl	Kno30-3fl-1rl	Kno30-3fl-2rl
$\max(d)$	196	173	172	180
$\max(f)$	9883	9644	9967	9968
$\text{corr}(f^1, f^2)$	0.0	0.4	0.4	0.7
$\text{corr}(f^1, f^3)$	—	—	0.0	-0.5
overlap	1.0	1.0	0.7	0.7
fd^1, fd^2, fd^3	206,260,—	243,315,—	235,354,252	234,324,342
dd	54.9	57.9	56.1	58.6
LS Measure				
$\text{ent}(U)$	0.999971	0.999971	0.999959	0.999962
$\text{ent}(O)$	0.910517	0.870681	0.938710	0.915543
$\text{diam}(U'_1, rs)$	16 (10^3)	16 (10^3)	23 (10^3)	24 (10^3)
$\text{diam}(O'_1, rs)$	16 (10^3)	16 (10^3)	23 (10^3)	23 (10^3)
$\text{lmm}(U'_1, rs)$	14.73 (10^3)	15.459 (10^3)	26.007 (10^3)	26.239 (10^3)
$\text{diam}(O'_1, \bar{\lambda})$	16 (10^3)	16 (10^3)	24 (10^3)	23 (10^3)
$\text{ent}(O'_1, \bar{\lambda})$	0.900 (10^3)	0.847 (10^3)	0.853 (10^3)	0.846 (10^3)
$\text{lmm}(O'_2, \bar{\lambda})$	13.719 (10^3)	15.511 (10^3)	25.625 (10^3)	26.866 (10^3)
$\text{diam}(O'_3, nds)$	14 (200)	14 (200)	22 (200)	22 (200)
$\text{ent}(O'_3, nds)$	0.489 (200)	0.252 (200)	0.743 (200)	0.746 (200)
$\text{lmm}(O'_3, nds)$	16.695 (200)	17.310 (200)	28.945 (200)	29.955 (200)

of the local optima obtained, as well as properties of the starting solutions, gives some insight into how correlated the optima are, and thereby hints at the difficulty of finding a global optimum. The same strategy is applicable to the mQAP, although complicated by the fact we need to try to find optima all over the PF.

Algorithm 1 shows our approach. We start by defining a set of evenly distributed scalarizing vectors Λ . For each of these ‘directions’ we repeat several LS runs from random start solutions. For each such LS run, we record the starting point, its multiobjective evaluation, the local optimum reached, its multiobjective evaluation, the scalarizing weight vector used in the LS, and the number of LS moves applied. We then analyse these records. First, we can just compare some properties of all starting solutions, U , and optima O : the entropy, the diameter and the number of LS moves performed on U to obtain O , $\text{lmm}(U) = \text{lmm}(O)$. Note that although entropy and diameter were defined for the QAP [1], these can be used without alteration on the mQAP. Next, we can try to investigate properties of the landscape perpendicular and parallel to the PF. To do this we construct samples (subsets) of a complete set of records (i.e. of O or U or $O \cup U$) such that all records in the sample satisfy some condition on them. Three different conditions, and combinations of these, seem promising. The first is to

Algorithm 1 Outline of a landscape analysis tool for the mQAP

```

1: Input:  $n; \Lambda; \#searches; \text{mQAP instance } I; \text{conditions } C; \text{measures } M$ 
2:  $U \leftarrow \emptyset$ 
3:  $O \leftarrow \emptyset$ 
4: for each  $\vec{\lambda} \in \Lambda$  do
5:   for  $i = 1$  to  $\#searches$  do
6:      $\pi \leftarrow \text{RandomPermutation}(n)$ 
7:      $\pi' \leftarrow \text{LS}(\pi, I)$ 
8:      $r(\pi) \leftarrow (\pi, \vec{C}(\pi), \#LSmoves, \vec{\lambda})$ 
9:      $r(\pi') \leftarrow (\pi', \vec{C}(\pi'), \#LSmoves, \vec{\lambda})$ 
10:     $U \leftarrow U \cup \{r(\pi)\}$ 
11:     $O \leftarrow O \cup \{r(\pi')\}$ 
12:   end for
13: end for
14: for each measure  $mre \in M$  do
15:    $\text{print } mre(U)$ 
16:    $\text{print } mre(O)$ 
17: end for
18: for each condition  $c \in C$  do
19:    $S \leftarrow \{r(\pi) \in T \mid c(r(\pi)) = \text{TRUE}\}, \text{ where } T = O \vee U \vee O \cup U$ 
20:   for each measure  $mre \in M$  do
21:      $\text{print } mre(S)$ 
22:   end for
23: end for

```

consider only optima that have been found using a similar LS direction, which we can specify using a target vector ξ and the following condition: $\sum_{j=1}^k (\lambda^k - \xi^k)^2 < \epsilon$, where ϵ is some small positive value. The second condition is to take only solutions of the same nondominated rank. In the simplest case this just means taking only the nondominated solutions (i.e., Pareto optimal from among those points found). The third condition is to specify some target objective vector and take only points nearby to this. By measuring the entropy and diameter of these subsets we can get a feeling for how similar solutions within these subsets are. We can of course combine these partitioning conditions together to further our investigations. We are also able to measure fitness-distance correlations within our different subsets using, as fitness, the Manhattan distance between points in the objective space.

Table 1 presents some illustrative results of our landscape analyses for five different mQAP instances. These were collected by running 1000 local searches from each of 100 (for 2-objective instances) or 105 (3-objective instances) different λ vectors, thus giving us approximately 200000 records in all. Although at this stage it is not possible to draw any conclusions about which overall search strategy would be most suitable for each instance, we can see that pertinent information is provided. For example, we can see that the diameter of the subsets tends to be about the same size as that of all of the entire sets U and O . Entropy, however, changes, indicating that although optima can be found over a large region, they are at least clustered together in groups. For the 2-objective instances, the entropy of the nondominated points is quite low. This means, particularly in the instance with positive correlation (0.4) in the flow matrix entries, that points on the nondominated front are quite similar.

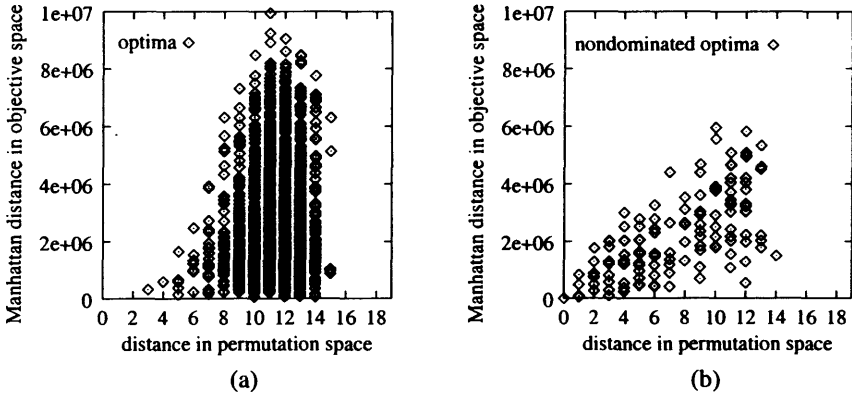


Figure 2: Fitness-distance correlation plots between all pairs of points. In (a) a random sample of local optima is plotted. In (b) a sample (of the same size) of nondominated optima is plotted. The greater fitness-distance correlation in (b) is clear

This suggests that searching parallel to the PF may be much more efficient than re-starting searches from random points. Points found using a particular λ vector, or one close to it, do not have such a low entropy. This might suggest that on the way towards the PF optima are spread all over the space and are only in relatively small clusters. Thus from the preliminary results seen here we might guess that it is easier to move along the PF than to get to the PF in the first place. Many more experiments using these techniques are needed to confirm this hypothesis and to make more detailed inferences or predictions.

In Figure 2 we can see two fitness-distance correlation plots for (a) all pairs of a random sample of optima and (b) all pairs of the nondominated optima of instance *Kno20-2fl-2rl*. Clearly, the nondominated optima are highly correlated, indicating that it is probably quite easy to find solutions on the PF once one has been found.

7 Conclusion

Some problems with multiple objectives can be searched using a hybrid approach which makes use of an efficient local search. When deciding on an overall search strategy, however, we argue that one must consider whether there is correlation between nearby optima or not. In a multiobjective landscape we can measure such correlations either ‘perpendicular’ or ‘parallel’ to the Pareto front. This will help us decide whether our search should first approach the PF and then spread around from there, or whether our search should start repeatedly from new random points, or whether it should use a gradual approach towards the PF from all directions in parallel. In this paper we have introduced and discussed these notions and given some methods for measuring the landscape of a multiobjective combinatorial optimization problem, mQAP, which we also introduced here for the first time.

Acknowledgments

Joshua Knowles gratefully acknowledges the support of a Marie Curie research fellowship of the European Commission, contract number HPMF-CT-2000-00992.

References

- [1] Vincent Bachelet. *Métaheuristiques Parallèles Hybrides: Application au Problème D'affectation Quadratique*. PhD thesis, Université des Sciences et Technologies de Lille, December 1999.
- [2] P. Czyzak and A. Jaskiewicz. Pareto Simulated Annealing. In G. Fandel and T. Gal, editors, *Multiple Criteria Decision Making. Proceedings of the XIth International Conference*, pages 297–307, Hagen, Germany, 1997. Springer-Verlag.
- [3] Éric D Taillard. A comparison of iterative searches for the quadratic assignment problem. *Location Science*, 3:87–105, 1995.
- [4] C. Fleurent et J.A. Ferland. Genetic hybrids for the quadratic assignment problem. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 16:173–188, 1994.
- [5] Carlos M. Fonseca and Peter J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.
- [6] Michael Pilegaard Hansen. Tabu search in multiobjective optimisation : MOTS. In *Proceedings of the 13th International Conference on Multiple Criteria Decision Making (MCDM'97)*, Cape Town, South Africa, January 1997.
- [7] Michael Pilegaard Hansen and Andrzej Jaskiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Institute of Computing Science, Poznan University of Technology, ul. Piotrow 3a, 60-965 Poznan, Poland, March 1998.
- [8] Andrzej Jaskiewicz. On the performance of multiple objective genetic local search on the 0/1 knapsack problem. a comparative experiment. Technical Report RA-002/2000, Institute of Computing Science, Poznan University of Technology, Poznań, Poland, July 2000.
- [9] T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In L. J. Eshelman, editor, *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 184–192. Morgan Kaufmann, 1995.
- [10] S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the ACM*, 23:555–565, 1976.
- [11] Thomas Stützle. Iterated local search for the quadratic assignment problem. Technical Report AIDA-99-03, Darmstadt University of Technology, Computer Science Department, Intellectics Group, 1999.
- [12] T. E. Vollmann and E. S. Buffa. The facilities layout problem in perspective. *Management Science*, 12(10):450–468, 1966.

Active Learning Using One-class Classification

Ibrahim Gokcen, Jing Peng, Bill P. Buckles
EECS Department, Tulane University, New Orleans, LA, 70118
{gokcen.jp,buckles}@eecs.tulane.edu

Abstract. Active learning aims at minimizing the number of labeled examples and at the same time reaching the optimum as fast as possible. In this paper, we propose a new parameterization for active learning, which is described based on the idea of one-class classification. We demonstrate the use of this parameterization by proposing a simple heuristic and a volume reduction metric for active learning. Empirical results on a variety of data sets show that our metrics outperform and are comparable with other proposed active learning metrics.

1 Introduction

Supervised learning techniques such as neural networks [6] and support vector machines [3] require a collection of tuples (\mathbf{x}_i, y_i) to learn the boundary that separates the classes or the characteristics of the input space. In each such tuple, \mathbf{x}_i ($i = 1, \dots, l$) represent the training examples of some dimensionality d and y_i ($i = 1, \dots, l$) represent the labels or classes (we assume $y_i \in \{-1, 1\}$) of the corresponding training examples, as provided by a teacher or an expert. l is the number of training examples. However, obtaining an accurate classifier requires a large amount of labeled data to be incorporated into the training set. Labeling is typically done manually and is expensive. Therefore we have two choices. We can either find alternative sources to reduce the need for labeled data, or we can label the data in a way that reduces the number of labelings. As an example to the first approach, Nigam uses expectation-maximization (EM) to estimate class labels for the unlabeled data and uses this information to increase classifier accuracy [10]. In this paper, we will choose the second approach, namely choosing the data to label appropriately. Labeled data influences the initialization of the classifier and this initial classifier plays an important role in approaching the optimal classifier. This second approach is commonly known as 'Active Learning'.

Active learning aims at minimizing the number of labelings until the stopping criterion has been met. In other words, it enables the learner to approach the desired solution faster. It uses the set of labeled examples L to define a model of the input space and then uses the unlabeled examples to incrementally improve the learner's model, until a satisfactory model of the input space can be defined. The choice of the examples in the unlabeled set UL (i.e. $\mathbf{x}_i \in UL$) that are to be labeled, plays an important role in achieving this goal. Effectiveness scores can be assigned to examples and based on those scores, unlabeled examples can be labeled and incorporated into the training set.

Researchers have developed several methods over the years to achieve active learning. Some works can mainly be characterized as simple heuristic approaches to the problem, that

employ the most-probable or the most-informative criterion [4, 13, 5]. These are mostly based on computing the distance of an example to the hyperplane and then assigning effectiveness scores according to that distance. Disadvantages of these approaches are demonstrated by Tong and Koller [19]. They tend not to explore the feature space aggressively and can end up ignoring entire clusters of unlabeled instances. On the other hand, greedy optimal strategies, which also can be considered as volume reducing approaches, include statistical models and mostly employ the most-informative criterion [2, 16]. They are more stable but also computationally more expensive. Active learning has also been used for estimating class probabilities by selecting most-informative examples to sample the space better [12]. These can also be considered as the examples, which are the most dissimilar to the examples already included and are likely to be outliers or positioned on the boundary [11]. Although lots of research have been done in the field over the years, principled approaches began to emerge only recently.

Tong and Koller [19] used the duality of the feature space (F) and the parameter space (W), to provide a principled approach for active learning using SVMs. Details about SVMs are given in [3]. In their work, version space V is defined as the surface area of the ball that defines the parameter space W (assuming we have constant $\|\mathbf{w}\|$), which includes all the weight vectors \mathbf{w} , that classify the training examples correctly. V can be formulated as follows

$$V = \{\mathbf{w} \mid \|\mathbf{w}\| = 1, y_i(\mathbf{w} \cdot \phi(\mathbf{x}_i)) > 0, i = 1 \dots l\} \quad (1)$$

Each unlabeled example $\phi(\mathbf{x}_i)$ (transformed to feature space by a mapping ϕ) divides the version space V into two half-spaces, and the boundaries of V are determined by the labeled training examples. The goal is to divide V in such a way that we reach the optimum solution \mathbf{w}^* in the fewest number of iterations. Version space only exists if the training data are linearly separable in the feature space. However, this isn't a significant restriction in practice. First of all, feature space generally has a very high dimension (possibly infinite) and this results in the data set being separable. Secondly, it's possible to modify any kernel, so that the data in the new induced feature space is linearly separable [19]. An example modification is given by Shawe-Taylor and Cristianini [17].

None of the approaches mentioned above has considered capturing feature space distributions to define space representations and measures. The problem with the feature space distributions come from the fact that we don't know the mapping ϕ and therefore we can not sample the feature space precisely. However, we can define ways to separate examples from each other and with this, we can define alternative spaces to use for active learning.

In this paper, we define a novel space, which we call the C space, and a novel way to assign effectiveness scores to the unlabeled examples. This space is defined by using the idea of one-class classification (OCC). OCC has been established and used by numerous researchers such as Chen et al. [1], Tax and Duin [18] and Scholkopf et al. [14] and is a generalization of the discriminating hyperplane in a non-linear way. Scholkopf et al. showed that finding the hyperplane, that maximizes the margin between the positive examples and the origin, is equivalent to finding the hypersphere that encloses all the positive examples and excludes the negative ones [14]. We show that this space representation and an appropriate effectiveness score assignment procedure provide us with a principled way to choose examples to label.

The paper is organized as follows. In Sect. 2, we describe the idea of OCC and explain the concept of C space using OCC. Section 3 gives two different types of metrics (simple

heuristic and volume reduction metric) that we can use for active learning. Empirical evaluation is given in Sect. 4 and the paper concludes with discussion of results and future research directions.

2 One-class classification using hyperspheres

One-class classification (OCC) [1, 14, 18] tries to estimate the distribution of the target examples in the feature space using a minimal radius hypersphere, that includes all the positive examples (with $y_i = 1$). More often that not, negative examples (with $y_i = -1$) are hard to identify and the only information we have are the positive ones. OCC estimates the support (the region where the probability density lives) that can include most of the positive examples, with some regularization to single out outliers [1]. It first maps the data into the feature space using a feature map ϕ , and then uses a hypersphere to describe the data in the feature space and put most of the data into that hypersphere. We can use a kernel K [15] that complies with Mercer's Theorem [8] to operate on the feature space. The simplest such K would be the linear kernel ($K(\mathbf{x}_i, \mathbf{x}) = \mathbf{x}_i \cdot \mathbf{x}$). Another kernel that we can use is the so-called Radial Basis (RB) kernel ($K(\mathbf{x}_i, \mathbf{x}) = e^{(\gamma\|\mathbf{x}_i - \mathbf{x}\|^2)}$). OCC can be formulated as

$$\min R^2 + \frac{1}{vl} \sum_i \zeta_i \quad (2)$$

$$s.t. \|\phi(\mathbf{x}_i) - \mathbf{c}\|^2 \leq R^2 + \zeta_i, \zeta_i \geq 0, i = 1, \dots, l$$

where R and \mathbf{c} are the radius and center of the ball that contains all positive examples and $\phi(\mathbf{x}_i)$ are the transformed examples in the feature space. ζ are also included as the regularization parameter to single-out outliers, as in the soft-margin SVM case. $v \in [0, 1]$ defines the tradeoff between the minimal R and maximum number of feature space examples that can fit into the hypersphere. If we formulate this problem using Lagrange multipliers and solve it, we get the following constraints

$$\sum \alpha_i = 1 \quad (3)$$

$$0 \leq \alpha_i \leq \frac{1}{vl} \quad (4)$$

$$\mathbf{c} = \sum_{i: y_i=1} \alpha_i \phi(\mathbf{x}_i) \quad (5)$$

From (5), it can be seen that \mathbf{c} is a linear combination of the feature space examples and it lies in the sub-space spanned by them, as does \mathbf{w} . α_i are computed by solving the quadratic optimization problem (2) and denote the importance of a feature space example $\phi(\mathbf{x}_i)$ in determining the center. For simplicity, we can assume that all examples have equal importance, by assigning $\frac{1}{l}$ to each α_i . With this assumption, instead of determining the center with (5), we take the mean of all the labeled positive examples that have been available up to that point. This simplification is a special case of the optimization with equal coefficients, eliminates the computationally expensive optimization process and still gives good results.

2.1 Alternative spaces

The enclosing hypersphere idea described in the previous section can be applied to find alternative spaces for active learning, with some modifications. As the SVM hyperplane discriminates two different classes linearly (in a linearly separable problem), the hyperspheres divide the feature space into two parts in a non-linear way, with the hypersphere boundaries themselves being the discriminants. If we eliminate the soft-margin constraint and consider all the hyperspheres, that include all positive examples but no negative ones, we can define the counterparts of W , V and \mathbf{w}^* , using the hypersphere definition.

In our alternative space, we will call S as the space of all hyperspheres, that can be defined in the feature space. Since each hypersphere can be defined by a tuple of center and radius $\langle \mathbf{c}, R \rangle$ (there exists a bijection between every tuple and hypersphere), S is

$$S = \{ \langle \mathbf{c}, R \rangle \mid \mathbf{c} \in \mathbb{R}^d \} \quad (6)$$

where d is the dimensionality of F (possibly infinite). Equation (6) is similar to the definition of the parameter space W and consequently to the definition of H [7].

The counterpart of the version space V can easily be found, if we consider the hyperspheres that include all positive examples and exclude the negative ones. We call the space of $\langle \mathbf{c}, R \rangle$ tuples that denote such hyperspheres as C . Remember that by definition, version space V only exists if the training data are linearly separable. In our case, such hyperspheres should exist in the feature space that separate positive examples from the negative ones. In the feature space, we cannot assume that the positive examples are clustered in spherical shape - images can have complicated non-linear distributions. However, by using kernel-based forms, non-linear distribution can be dealt with in the same framework [1]. In the case of RB kernel, $K(\mathbf{x}_i, \mathbf{x}_j) > 0$ for all i, j , thus all inner products between mapped patterns are positive, implying that all patterns lie inside the same orthant. Moreover since $K(\mathbf{x}_i, \mathbf{x}_i) = 1$ they all have unit length. Hence they're separable from the origin and equivalently they can be enclosed inside a hypersphere [14]. Therefore, such hyperspheres are guaranteed to exist in the feature space if we use RB kernels. Since all the positive examples are inside the hypersphere and all the negative ones are outside, this can be represented with

$$\|\phi(\mathbf{x}_i) - \mathbf{c}\|^2 \leq R^2 \text{ if } y_i = 1 \quad (7)$$

$$\|\phi(\mathbf{x}_i) - \mathbf{c}\|^2 > R^2 \text{ if } y_i = -1 \quad (8)$$

Equations (7) and (8) can be represented in a single inequality as

$$y_i(\|\phi(\mathbf{x}_i) - \mathbf{c}\|^2) \leq y_i R^2 \quad (9)$$

Thus, the definition of C becomes

$$C = \{ \langle \mathbf{c}, R \rangle \mid y_i(\|\phi(\mathbf{x}_i) - \mathbf{c}\|^2) \leq y_i R^2, \forall i \} \quad (10)$$

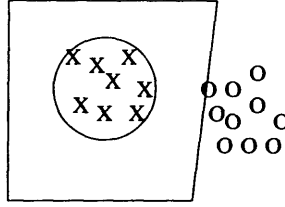


Figure 1: Polyhedron and minimal radius hypersphere

where d is the dimensionality of F and $\langle c, R \rangle \in \mathbb{R}^{d+1}$.

We can illustrate the concepts of the previous paragraph in Fig. 1. In the figure the volume between the polyhedron, whose one side is specified by the hyperplane (with normal w^-), that passes through the negative support vectors ($w^- \cdot x + b = -1$) and the other sides are specified by feature space boundaries (can be infinite but for the purposes of illustration we assume that such boundaries exist) and the minimal radius hypersphere defines the volume where positive examples can lie (except for the outliers). Therefore, feasible hyperspheres (defined by Eq. 10) lie in this volume. Being in C ensures that the hyperspheres that are still feasible contain all the positive examples that have been seen up to the current iteration. This set of hyperspheres has to contain the minimal radius hypersphere (MRH) (also shown in Fig. 1) and have to be inside the polyhedron. Therefore we can call the volume between the polyhedron and the MRH as the feasible region. With the addition of every negative example that's within the margin, the polyhedron and the feasible region shrinks. With the addition of positive examples, MRH expands and thus the feasible region again shrinks. Therefore, addition of new examples result in C spaces of the form $C_1 \supset C_2 \supset C_3 \dots$ and the optimum hypersphere $\langle c^*, R^* \rangle$ (MRH had we seen all examples) always stays within the feasible region. This shows that the counterparts of W , V and w^* are S , C and $\langle c^*, R^* \rangle$ respectively. In the next section we will propose metrics for active learning using the idea of C space. One important contribution of this paper is the feasible region definition given above. We will use this formulation in proposing a volume reducing metric in Sect. 3.2. Version space techniques approach the optimum classifier by reducing the version space as much as possible at each iteration. Our goal is to reduce the feasible region (in the feature space) using our hypersphere parameterization.

3 Active Learning in C space

Having defined our S and C spaces, we can define how active learning would be achieved. The goal of active learning in pattern classification is to approach the optimum classifier (Bayes classifier) as fast as possible, using as small number of training examples as possible. In our case, we would like to choose examples, that enable us to come as close to the optimum hypersphere as possible. In this section, we present two different types of metrics to do that: A simple heuristic metric and a volume reduction metric. Examples are ranked according to their effectiveness scores that are based on these metrics.

3.1 Simple heuristic metric

Simple heuristics usually involve the computation of dot products which can be kernelized. In this section we propose such a metric for active learning using \mathbb{C} space parameterization and later in Sect. 4, we compare our metric with two other metrics. We assign the effectiveness scores to unlabeled examples in a fashion that is similar to the approach of Chen et al. [1]. The difference of our approach is in the effectiveness score assignment. Chen et al. assigns larger effectiveness scores to the examples that are closer to the center (therefore the most-probable ones). In our approach we use the distance to the center as a unified framework for choosing both most-probable and most-informative examples. We first choose our examples that are most distant from the center (explore the feature space) and then choose examples that are close to the current center (exploit the information we get about the feature space). This heuristic is similar to the one proposed by Tong and Keller [19]. To achieve this, we have to map all the examples to a high-dimensional space by using a kernel. The kernel function K is sufficient to compute distance to the current center. To ensure that positive examples can be enclosed inside a hypersphere in the feature space, we use the RB kernel. By changing the values of γ , we can adjust the kernel to satisfy our needs. If we represent the distance by D , the squared Euclidean distance between the current center \mathbf{c} and an unlabeled example $\phi(\mathbf{x}_i)$ in the transformed space is given by

$$D = \|\phi(\mathbf{x}_i) - \mathbf{c}\|^2 = (\phi(\mathbf{x}_i) - \mathbf{c}) \cdot (\phi(\mathbf{x}_i) - \mathbf{c}) \quad (11)$$

Using (5) to compute \mathbf{c} , kernels instead of the dot products and the fact that if K is the RB kernel then $K(\mathbf{x}_i, \mathbf{x}_i) = 1$, we end up with the following distance formulation

$$D = \sum_{j,k} \alpha_j \alpha_k K(\mathbf{x}_j, \mathbf{x}_k) - 2 \sum_{j,i} \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) + 1 \quad (12)$$

Here, labeled positive examples are indexed by k and j and unlabeled examples are indexed by i . As discussed in Sect. 2, for simplicity, we can assign equal importance to all examples and thus, we can drop α_i 's from the equations to get

$$D = \left(\frac{1}{l_+}\right)^2 \sum_{k,j} K(\mathbf{x}_k, \mathbf{x}_j) - \frac{2}{l_+} \sum_{j,i} K(\mathbf{x}_i, \mathbf{x}_j) + 1 \quad (13)$$

where l_+ denotes the number of positive examples that are used to compute the current center (i.e. that have been labeled). In the equation we can see that there is one variable component and two constant terms. Constant terms are the same for all unlabeled examples. However, the variable component (with $K(\mathbf{x}_i, \mathbf{x}_j)$) changes depending on the similarity between the unlabeled examples and the positive examples that are used to compute the center. To have a smaller distance, the similarity between a positive and an unlabeled example have to be big. Therefore the cases where the unlabeled examples are similar to the positive examples result in smaller distances than others.

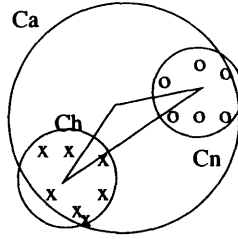


Figure 2: Three hyperspheres

3.2 Volume reduction metric

One other and better way to approach the optimum hypersphere faster is to reduce the volume of the feasible region (defined in Sect. 2.1) that the hyperspheres can be in, as fast as possible. This is similar to the idea of version space reduction. In higher dimensions it's hard to compute the volume of the feasible region. However, we can approximate this volume by using the concepts we have already established.

The boundaries of the feature space (polyhedron boundaries in Fig. 1) can be approximated by a hypersphere that contains all the labeled examples seen up to that point. Let's represent the center and radius of this hypersphere H_a with c_a and R_a . We also have the MRH H_p . Its center and radius are c_p and R_p . Same kind of hypersphere H_n exists for the negative examples with center c_n and radius R_n . The volume that we want to minimize when adding a new example is roughly the difference between the volumes of these three hyperspheres. If the volumes can be computed, this difference is $D = V_a - V_p - V_n$ where V 's denote the volumes of the corresponding hyperspheres. Our first observation is that this volume can be approximated roughly by taking into account the fact that when the centers are closer to each other, the volume difference would be less. One way to incorporate center information into the measure is to consider the triangle whose vertices are the three centers (unless three points are on the same line). The smaller this triangle is, the smaller the volume difference is approximately. Figure 2 shows the intuition behind this observation in the feature space.

One easy way to approximate the size of the triangle is to compute the perimeter. Perimeter is simply the sum of all distances between the three centers which is

$$D = \|c_p - c_n\| + \|c_p - c_a\| + \|c_a - c_n\| \quad (14)$$

If we use the square of each distance instead of itself and use the dot product to compute these squared distances, we can kernelize this distance.

By using an RB kernel, using dot products instead of magnitude squares (as in (11)) and computing centers with simple averaging, the sum of the squared-distances becomes

$$D = c_1 \sum K(\mathbf{x}_i, \mathbf{x}_i) + c_2 \sum K(\mathbf{x}_j, \mathbf{x}_j) - c_3 \sum K(\mathbf{x}_i, \mathbf{x}_j) \quad (15)$$

where $c_1 = \frac{2}{a} - \frac{2b}{l^2}$, $c_2 = \frac{2}{b} - \frac{2a}{l^2}$, $c_3 = \frac{4}{ab} - \frac{4}{l^2}$, i and j index positive and negative examples and a and b denote the number of positive and negative examples respectively. For each unlabeled examples, we first label the example as positive and compute the distance D^+ . Then we label

it as negative and compute the distance D^- . Determining which unlabeled example reduces the volume most can simply be done by choosing the example that minimizes $\frac{D^+ + D^-}{2}$. Other ways also exist such as choosing the example which minimizes the minimum of the two (D^+ and D^-) or the minimum of their ratios.

4 Empirical Evaluation

To evaluate the performance of different methods we use four different data sets. The methods are Chen et al.'s OCC (OCC) with averaging, simple heuristic OCC given in Sect. 3.1 (OCC-MI), simple heuristic method given by Tong and Koller (TONG) and the volume reduction method given in Sect. 3.2 (OCC-VOL). We use two data sets from UCI machine learning repository [9] (namely sonar and vote) and two simulated data sets. For the simulations, we created 100 examples (of dimensionality 2) for each class using different means for each dimension (μ_1 and μ_2). Moreover, we played with the first two eigenvalues of each distribution (λ_1 and λ_2) to end up with different distribution shapes for different classes. Means and eigenvectors constitute parameter sets $P_i = \{\mu_{j1}, \mu_{j2}, \lambda_{j1}, \lambda_{j2}\}$ where j indexes the classes. We generated two different data sets with four classes using two such parameter sets (P_1 and P_2). P_1 (Fig. 5) has overlapping and elliptic cluster of examples whereas P_2 (Fig. 7) has circular and less overlapping cluster of examples. Sonar (of dimensionality 60) and vote (of dimensionality 16) data sets both have two classes with 208 and 232 examples respectively.

For each data set we assume that we start with a positive example (query) which is chosen randomly from the data set and assumed to be positive. Since we need at least one negative example for TONG, we choose the example that is most distant (in Euclidean distance) from the query and assume that it's negative. This assumption turned out to be correct for almost all experiments. Precision values are computed at each iteration (total of eight iterations) and are given in Figures 3, 4, 6 and 8. Effectiveness scores for all unlabeled examples are computed and they are ranked based on their scores. First eight are returned to be labeled.

For all methods except for OCC we first do four iterations to explore the space (choose examples that are most-informative) and then report the precision values from the next four iterations. For OCC we just report the results from the last four iterations. This is done for each class in the data sets and the precision values are averaged over the number of classes to make the results statistically significant. Furthermore, we conduct the same experiment 10 times to reduce the effect of the initial choice of example on the final solution. As in all similar methods, initial model plays an important role in approaching the optimum solution. Kernel parameters are chosen by cross-validation. Results show that OCC-MI and OCC-VOL perform well for all the data sets. OCC-MI especially outperforms the other methods in all the data sets except for vote data. Even in that case at the final iteration it matches the best method for that data set (TONG). OCC-VOL is a rather crude approximation of the volume reduction and is outperformed in general. However, because of the way it selects unlabeled examples, it has the steepest precision increase in most cases and at the last iteration catches up with the other methods. One other observation is the fact that, in high-dimensional data sets (such as sonar data), OCC-VOL performs better than in low-dimensional data sets. This also proves that the performance of simple heuristic methods are degraded more in higher dimensions, whereas volume reduction methods still perform well. Also note that the metric given in (15) is a rough approximation of the volume that we actually want to compute. However, it still gives comparable results with the simple heuristic methods in all data sets.

From the simulated data sets we can further see that all methods have increased precision in the case where classes are more separable and OCC-MI is the best in terms of precision. In the non-spherical input distribution case (simulated data 1), OCC-VOL performs worse than the spherical case. This actually is an expected result considering the fact that it tries to approximate the volume by using spherical shapes. For the spherical data case (simulated data 2), it's the second best metric in terms of precision at the final iteration. One other advantage of OCC-VOL is the fact that it does not include any optimization computation, which makes it computationally as efficient as simple heuristic metrics. In general, this is not true for other volume reduction methods.

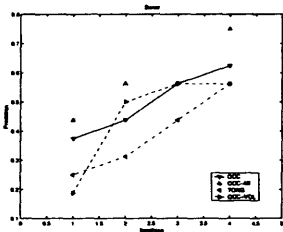


Figure 3: Sonar data

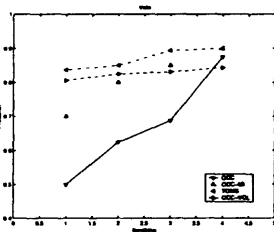


Figure 4: Vote data

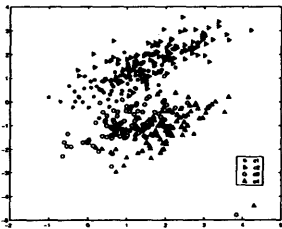


Figure 5: Simulated data 1

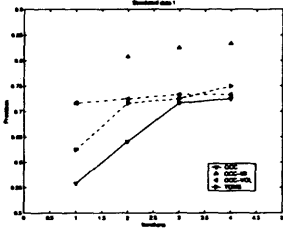


Figure 6: Simulated data 1 results

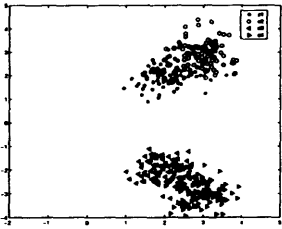


Figure 7: Simulated data 2

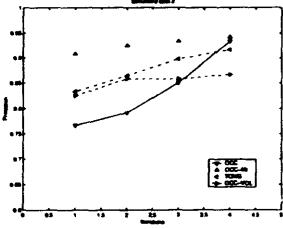


Figure 8: Simulated data 2 results

5 Conclusion

In this paper we presented a novel parameterization for active learning and two novel metrics for effectiveness score assignment, based on the idea of one-class classification. The metrics we propose provide us with ways to do faster and more accurate active learning. Results show that the space representation we propose enables us to capture the feature space distribution

better and can lead to more accurate metrics than the ones proposed by using other space representations.

As a future work, we intend to approximate the volume of the feasible region in a better way to obtain more accurate metrics. We also intend to theoretically analyze ways to determine if an arbitrary hypersphere is in the C space (thus lies in the feasible region) or not. This would enable us to gain additional insights into the properties of the feasible region that we are interested in.

References

- [1] Y. Chen, X. S. Zhou, and T. S. Huang. One-class svm for learning in image retrieval. In *Proceedings of IEEE International Conference on Image Processing*, pages 34–45, 2001.
- [2] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Advances in Neural Information Processing Systems*, 7:705–712, 1995.
- [3] N. Cristianini and J. Shawe-Taylor. *Support Vector Machines*. Cambridge University Press, 2000.
- [4] H. Drucker, B. Shahrory, and D. C. Gibbon. Relevance feedback using support vector machines. In *Proceedings of the 2001 International Conference on Machine Learning*, 2001.
- [5] Y. Freund, H. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2/3):133–168, 1997.
- [6] S. Haykin. *Neural networks, a comprehensive foundation*. Prentice Hall, 1994.
- [7] R. Herbrich, T. Graepel, and C. Campbell. Bayes point machines. *Journal of Machine Learning Research*, 1:245–279, 2001.
- [8] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Royal Soc. London, A* 209, pages 415–446, 1909.
- [9] C. J. Merz, P. M. Murphy, and D. W. Aha. Uci repository of machine learning datasets - uc irvine, 1997.
- [10] K. P. Nigam. *Using unlabeled data to improve text classification*. PhD thesis, CMU, 2001.
- [11] E. Pekalska, P. Paclik, and R. P. W. Duin. A generalized kernel approach to dissimilarity-based classification. *Journal of Machine Learning Research*, 2:175–211, 2001.
- [12] M. Saar-Tsechansky and F. Provost. Active learning for class probability estimation and ranking. In *IJCAI*, pages 911–920, 2001.
- [13] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *Proc. 17th International Conf. on Machine Learning*, pages 839–846, 2000.
- [14] B. Scholkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.
- [15] B. Scholkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- [16] H. S. Seung, M. Oppor, and H. Sompolinsky. Query by committee. In *Proceedings of Computational Learning Theory*, pages 287–294, 1992.
- [17] J. Shawe-Taylor and N. Cristianini. Further results on the margin distribution. In *Computational Learning Theory*, pages 278–285, 1999.
- [18] D. M. J. Tax and R. P. W. Duin. Data domain description by support vectors. In *Proceedings of European Symposium on Artificial Neural Networks 1999*, pages 251–256, 1999.
- [19] S. Tong and D. Koller. Support vector machine active learning with application to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.

Enhancing Real-World Applicability by Providing *Confidence-in-Prediction* in the XCS Classifier System

P.W.Dixon, D.W.Corne, M.J.Oates

*Department of Computer Science. University of Reading, Reading, RG6 6AY, UK
pwdixon@reading.ac.uk, d.w.corne@reading.ac.uk, moates@btinternet.com*

Abstract. Classifier systems are machine learning ruleset-discovery systems; the XCS classifier system has been found in recent years to compete well with rival machine learning systems on difficult benchmarks, and is now being intensively researched for real world applications. A problem common to all such systems is their accuracy on unseen data, and hence their real-world performance. This is not surprising, and standard methods such as cross-validation and early-stopping are commonly used in training to assess likely performance on unseen data. However, an additional and related issue is the confidence we can have in a prediction as a function of our confidence in the inputs. Predictions which lay on the boundary between two differing outcomes (e.g. the system may say 'malignant' in response to an input pattern, but a slight difference in that pattern might cause it to respond 'benign') must somehow be identified and questioned on their validity. We describe a technique which takes a ruleset learned by XCS (or another system), and provides highly useful confidence information when predictions are made with that ruleset. Further, we describe a measure which enables confidence-in-prediction behaviour to be assessed for different rulesets. Using this measure, called zero tolerance performance, we find that small, succinct and general rulesets produce better confidence-in-prediction performance. This mitigates against direct use of classifier system learned rulesets, since these tend to be very large, but it validates recent research which is successfully applying post-processing techniques which quickly reduce such a ruleset to a much smaller but equally accurate one. Development and testing is done on the standard Wisconsin Breast Cancer database.

1. Introduction

A Learning Classifier System (LCS) is a machine learning system in which a collection of rules (classifiers) are learned using a combination of reinforcement based learning and evolutionary computation (Holland, [6]; Goldberg, [4]; Lanzi et al, [8]). Originally described in a paper by Stewart Wilson [9], XCS is an LCS which works in a particular way, and has in recent times been found to be competitive with rival machine learning methods on benchmark machine learning datasets (Hartley, [5]; Wilson, [11]; Dixon et al, [2]). Consequently, there is continuing research in the capability of XCS and variants of it, with a view to real-world deployment as a general data mining and prediction tool.

In this paper, we focus on a particular aspect of machine learning systems which is of crucial relevance to real-world applications. This is confidence in prediction; that is, when a trained ruleset, or a learned decision tree, or a trained neural network (and so on) makes a particular prediction in a real-world scenario, how confident can we be in the accuracy of that prediction? The natural response is to refer back to benchmark results, cross-validation tests, and so forth, which yield estimates of the prediction accuracy for real-world use. For example, if the mean 10-fold cross-validation result for our decision tree learning method on

a large dataset was 74% accuracy, then we would by default assume a confidence of around 74% in the decisions it made on unseen data.

However, if we look more deeply into the performance figures and their relationship to the dataset, we would tend to find that the inaccurate predictions tend to arise from input patterns which are close to category boundaries, while input patterns more distant from such boundaries, and nested safely in the region of many other input patterns of the same category, tend to yield robustly accurate decisions. These are of course generalisations, but are accurate enough to form the basis of an approach towards providing differential confidence in predictions based on the input pattern under study.

In this paper, we describe a technique which can be used within a working rule-based prediction system to straightforwardly provide very useful extra information with each prediction it makes on unseen data. This enables a real-world deployment of machine learning systems such as XCS to be considerably more useful and liable to be better trusted and more readily taken up by potential users. We also provide a way to evaluate the confidence-in-prediction behaviour of any ruleset (or indeed any model learned from data), and use this to establish the relative real-world usefulness of rulesets with different sizes and generalities (though with equivalent accuracy performance on test data). All development and testing is done on the standard Wisconsin Breast Cancer dataset.

The remainder of this paper proceeds as follows, in section 2 we describe classifier systems and XCS in broad terms, and then describe the method used to provide confidence-in-prediction when a learned ruleset is used to make predictions. Section 3 then outlines a performance measure which is able to measure confidence-in-prediction behaviour for different rulesets. This is particularly interesting since the size and generality of rulesets, and the relation of such to accuracy and generalisation performance, is an issue of much research in classifier systems (Lanzi, [7]; Wilson, [10]), and it is of interest to establish further ways to distinguish between ruleset performance. Section 4 presents results comparing confidence-in-prediction behaviour over a range of learned rulesets (mainly with 100% performance accuracy) of different sizes and generalities. A concluding summary is given in section 5.

2. Classifier Systems and Providing Confidence in Prediction

A simplified description of the operation of the classifier system in the production of a ruleset begins by training input patterns being ‘covered’ by the creation of classifiers (rules) matching them. Covering is a process whereby, for every specific value in a field in the input pattern, an interval is produced in the corresponding field of the covering classifier which contains the input value. Such covering is done with a random amount of generality – that is, the input values are ensured to be covered, but with varying margins.

On subsequent training cycles, specific inputs are potentially matched by multiple classifiers which then compete to determine a predicted outcome. Comparison of the predicted and actual input outcomes allows modifications to be made to various classifier weightings which then direct the classifier set to become increasingly accurate in its predictions and more general in its applicability. During training the classifier population is operated on by an evolutionary algorithm which attempts to push the classifier set to include the more general rules which not only operate on the trained inputs but also provides a prediction of new inputs not in the training set. Goldberg [4] and Butz & Wilson [1] give fuller descriptions of the operation of classifier systems and XCS, respectively.

A straightforward implementation of XCS (and other such) systems for real-world use would provide a way to produce predictions, but no vehicle for establishing confidence in those predictions. Consideration of real-world scenarios suggests that users need such

confidence measures in order for the system to be considered for adoption. In common with all other applications based on any technology an initial period of confidence building is required, though this proving period may be more extended with systems based on experience/knowledge. This initial period overcomes users natural fears of application reliability and allows confirmation of results generated. In a system which initially does not necessarily have the 'full-story' on all possible input situations and outcomes this proving period may be especially long. Having a measure of confidence in the predictions made will go some way to assuring users of application reliability and returns to the user some of the control normally associated with these predictions hopefully assisting in the reduction of the proving period.

Our current implementation of XCS learns a ruleset from data, and then applies a ruleset reduction algorithm (Wilson, [12]; Dixon et al, [3]) to considerably reduce the size of the learned ruleset without altering its performance accuracy (as it turns out, this reduction step is helpful in terms of confidence-in-prediction performance, but not necessary to engineer confidence-in-prediction in the first place). Figure 1 shows a screenshot, indicating how confidence-in-prediction is presented to the user.

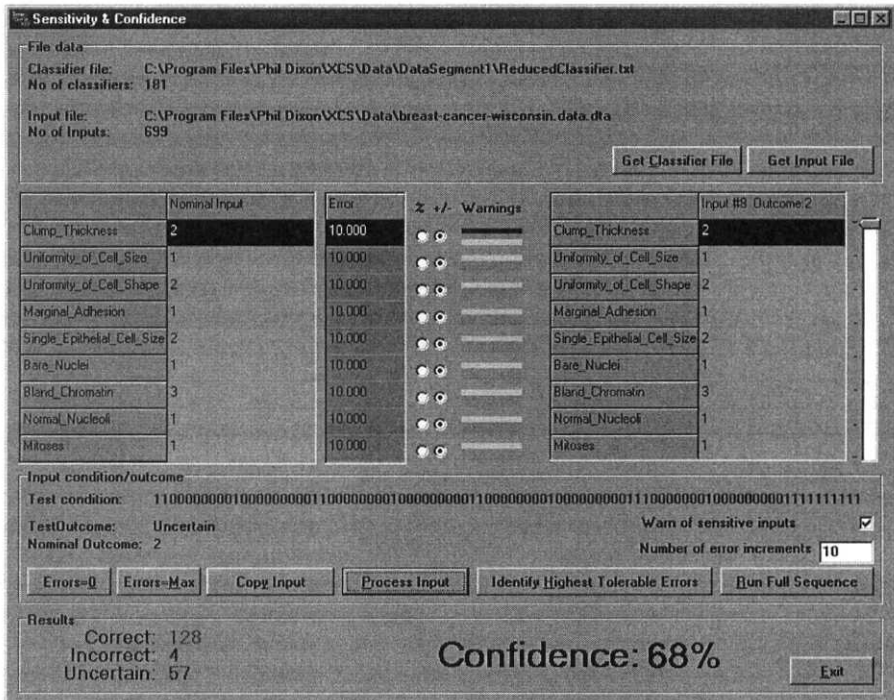


Figure 1 Application screenshot, showing how confidence-in-prediction is presented.

The confidence-in-prediction figure is generated whenever the system is asked to make a prediction based on an input pattern. The straightforward prediction ('nominal outcome' in the lower left of Figure 1) is 2, which indicates 'malignant' in this case. The confidence figure is calculated on the basis of the process in Figure 2, which applies a number of *tests* after finding the prediction for the particular input pattern under study (the 'nominal' input). Each test simply finds the ruleset's prediction for a slightly perturbed version of the input. The number of tests carried out depends on the number of errors increments being

tested (which can be set by the user individually for each variable), which should relate to the precision of and/or confidence in the input values themselves.

Tests can be performed singly simply to generate an outcome prediction at fixed preset error levels. Alternatively, tests can be performed with all error increments resulting in a display of maximum tolerable error on every variable field before confidence is decreased from 100% (relative to nominal input), see figure 2.

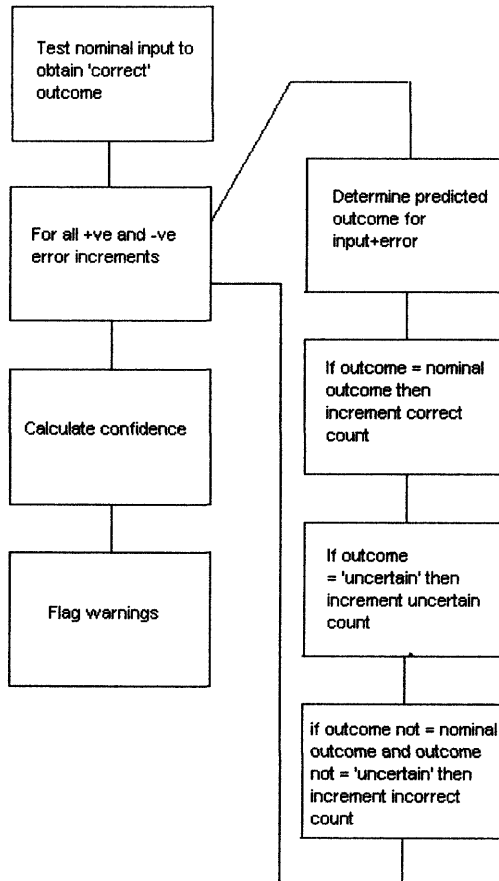


Figure 2 The incrementing error test process for providing confidence-in-prediction value when a prediction is generated from a ruleset

In all cases a traffic-light system of warnings is displayed with three bands for each variable indicating correct prediction (green), uncertain prediction (amber) and incorrect prediction (red) for nominal, and both +ve and -ve error increments.

3. Evaluating the Performance of Confidence-in-Prediction Outputs

The value providing confidence-in-performance information is intuitively clear, especially in real-world scenarios. Users are able to interact meaningfully with the system and work

with it over time to establish greater confidence in the predictions made by the system. However, the confidence information provided itself needs to be evaluated. We describe a suitable measure of performance as follows, based on analysing the results in terms of the highest 'tolerable' errors produced when all input data is tested. This allows us to compare different rulesets against each other in terms of their confidence-in-prediction performance, and we later describe experiments along those lines, which examine the relative confidence we can assign to rulesets of different sizes. This is of particular interest to us, since an ongoing line of research is, as mentioned above, the application of ruleset reduction algorithms in post-processing XCS rulesets to produce smaller, readable rulesets with similar performance.

We now explain how we produce a performance measure for confidence-in-prediction behaviour. For a given input pattern, the 'best' result would be a situation in which all inputs at all error tolerance levels led to the correct predicted outcome; i.e., all predictions in the 'error tolerance neighbourhood' of an input pattern match the outcome predicted for the unperturbed version of the input. We would be as sure as we could be of the prediction in such a case. A 'worst' result would be when, for a given input pattern, all of the perturbed patterns in its neighbourhood give a prediction different to that of the precise pattern.

For a given set of inputs, we would expect rulesets (or models in general) to vary in the degree to which such maximally error-tolerant and error-intolerant behaviour occurs. Our performance measure is aimed at measuring this across a set of inputs, specifically by measuring the number of maximally low-tolerance variables (a single input field, where any change in that input field changes the predicted outcome) in the inputs tested, leading to the following Zero Tolerance Performance (ZTP) measure:

$$\text{Zero Tolerance Performance (ZTP)} = \left(1 - \frac{\text{Total Number of Zero Tolerance Variables}}{(\text{Num. Test Inputs} \times \text{Num. Variables} \times \text{Total Num. Error Levels})} \right)$$

In the case of the WBC dataset and full input set testing, the number of test inputs is 699, the number of variables is 9 and the default setting for total number of error levels was chosen as 20 (10 +ve and 10 -ve error steps, equal to $1/10$ th the range of the variable; note: where the addition of the error causes the variable to exceed its upper/lower limit then the upper/lower limit value is used).

Using this definition of ZTP removes dependence on the actual number of error levels processed, however, of course, the higher the number of error levels the more reliable the resulting ZTP value. The total number of zero tolerance variables is the sum of the number of variables with zero tolerance resulting from predictions for all possible outcomes and for 'uncertain' outcomes. 'Uncertain' outcomes arise when the matchset of classifiers created for the presented input does not have a definite prediction.

In a 'real-world' situation the presence of a zero tolerance variable would be flagged to warn that a sensitive value is being used in the prediction and perhaps a check should be made or a more precise reading required to give confidence in the prediction.

A more representative indicator of the ZTP performance of a classifier set can be produced by a variation on the ten fold cross-validation process. Ten fold cross-validation is used to determine and expose any gross errors or assumptions when new 'unseen' situations are presented. For an explanation of the use of ten fold cross-validation in particular applied to the Wisconsin Breast Cancer dataset see Wilson [11]. In this case $9/10$ ths of the input data is trained to produce a 100% performing classifier set when tested on the remaining $1/10$ th of the input data.

4. Results

Using a standard XCS configuration (Butz & Wilson, [1]), with R_0 (the step-size used when generating a covering classifier) as a generality control, a number of classifier sets were produced by training on all input data and testing on all input data, the classifier sets being finalised when 100% performance is achieved. R_0 is the setting for the range of a covering classifier, e.g. an specific input field value of 5 may have a cover between 3 and 7 in which case R_0 could have been chosen as 2, (the value applied is in fact a randomised variable and can range between 1 and the setting of R_0). The 100% performance classifier sets were then reduced (Wilson, [12]; Dixon et al, [3]), and processed to establish ZTP values.

Two databases were chosen to illustrate the ZTP, the Wisconsin Breast Cancer Database and the Iris Plant Database.

Table 1 Zero tolerance levels at differing generalities

R_0	Iris Plants Database			Wisconsin Breast Cancer Database		
	Reduced classifier set size	Average population generality	Zero Tolerance Performance (ZTP)	Reduced classifier set size	Average population generality	Zero Tolerance Performance (ZTP)
1	80	0.0861	95.26	342	0.0888	97.09
2	48	0.1635	95.70	292	0.1655	97.95
3	45	0.2174	95.71	245	0.2394	98.73
4	28	0.3148	96.16	214	0.3071	98.70
5	31	0.3511	96.57	157	0.3739	99.23
6	31	0.3848	96.30	135	0.4223	98.98
7	24	0.4418	97.01	106	0.4658	99.39
8	25	0.4605	96.85	77	0.5143	99.54
9	25	0.4761	96.82	70	0.5606	99.49
10	22	0.4679	97.02	66	0.5624	99.58

The relationship between classifier average population generality and ZTP is positively correlated, although diminishing returns in ZTP are strongly apparent as we approach 60% generality. Interestingly, the graph of reduced classifier set size against generality shows a strong, and intuitively right, positive correlation between high generality and ‘succinctness’ of the ruleset. This reinforces the expectation that more general classifiers might be found to cover nearby inputs, and reduce the requirement for separate classifiers.

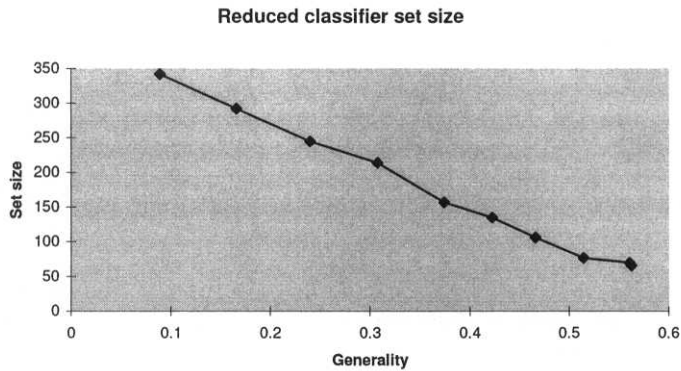


Figure 3 Relationship between reduced classifier set size and average population generality for Wisconsin Breast Cancer Database

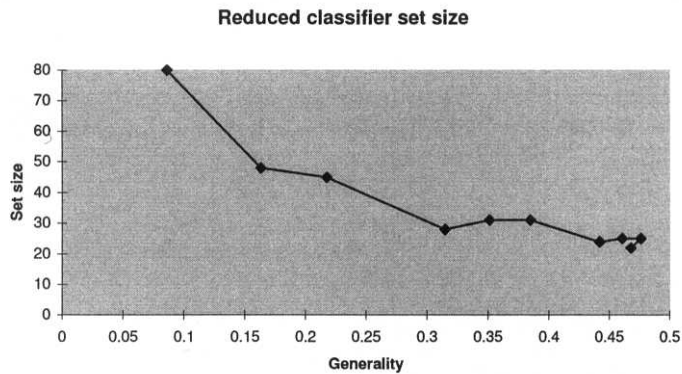


Figure 4 Relationship between reduced classifier set size and average population generality for Iris Plant Database

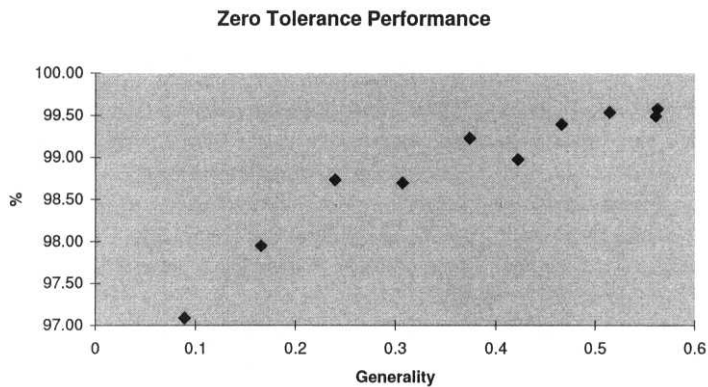


Figure 5 Performance graph for 100% training and test for Wisconsin Breast Cancer Database

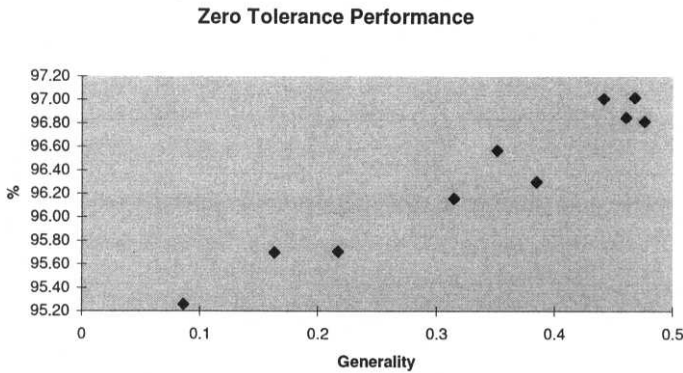


Figure 6 Performance graph for 100% training and test for Iris Plant Database

Ten fold cross-validation tests performed on the WBCD with a setting of $R_0=5$ produced the following results for all data folds. The results for ten fold cross-validation tests are lower in performance terms than those for the fully '100%-trained' system, which is also in line with expectations as performance can only be adversely affected by reducing the breadth of the training set. In a 'real-world' application experience and knowledge would be accumulated over time and holes in the knowledgebase would gradually be filled.

Table 2 ZTP for ten-fold cross validation tests on the WBC data

Test data fold	Average population generality	Zero Tolerance Performance (ZTP)
1	0.3525	96.83
2	0.3587	96.40
3	0.3605	96.87
4	0.3590	97.65
5	0.3537	97.55
6	0.3581	97.28
7	0.3681	96.87
8	0.3600	96.62
9	0.3648	97.30
10	0.3606	97.71
Average	0.3596	97.11

5. Conclusion

XCS is a machine learning system which produces rulesets, and which has recently been found to perform very well on benchmark machine learning problems, triggering much recent research into its general applicability as a real-world data mining/prediction tool. We have concentrated here on a previously little-explored (at least in the context of classifier systems) area relating to the real-world use of XCS as a prediction system, that of providing confidence-in-prediction in a way that enhances its real-world applicability. We describe

how a straightforward process can accompany the generation of a prediction, yielding useful confidence information about that prediction, and its sensitivity to particular input values. Further, we looked at how to evaluate particular rulesets in terms of the confidence information that they would typically yield. This involved defining a measure called Zero Tolerance Performance (ZTP), which tests the degree to which the ruleset will predict robustly. We found that ZTP performance correlated well with high generality, and small size, of the ruleset. This fits in well with current research on ruleset reduction methods for XCS, providing an additional reason (rather than just 'readability') to prefer smaller rulesets.

While this particular implementation uses a ruleset (classifiers) derived from an XCS based system, the results are applicable to any machine learning system which generates rulesets, and hence any prediction system which uses rulesets as the predictive model. Naturally, the techniques are also applicable to any dataset comprising numeric variables, and there are evident ways in which the confidence-in-prediction approach and performance measures analogous to ZTP can be derived for data containing symbolic variables, or a mixture of types of variable.

A natural extension in future work is to widen the error-tolerance testing to combinations of variables, and also to provide more sensitive performance measures than ZTP, which may be useful to distinguish between rulesets, or models in general, when their ZTP performance is similar.

Acknowledgements

The authors wish to thank BT Exact Plc for ongoing support for this research and Evosolve (UK registered charity no. 1086384) for additional support of this work.

References

- [1] Butz, M.V. and Wilson, S.W. (2000), 'An Algorithmic Description of XCS', Technical Report 2000017, Illinois Genetic Algorithms Laboratory, IL, USA.
- [2] Dixon, P.W., Corne, D.W., Oates, M.J. (2002) 'A Preliminary Investigation of Modified XCS as a Generic Data Mining Tool', in Lanzi, Stolzmann, Wilson (eds), *Proceedings of the 4th International Workshop on Learning Classifier Systems*
- [3] Dixon, P.W., Corne, D.W., Oates, M.J. (2002b), 'A Ruleset Reduction Algorithm for the XCS Learning Classifier System', in Lanzi, Stolzmann, Wilson (eds), *Proceedings of the 5th International Workshop on Learning Classifier Systems*, to appear.
- [4] David E. Goldberg (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Mass., 1989
- [5] Adrian Hartley (1998) Genetics Based Machine Learning as a Model of Perceptual Category Learning in Humans. Master's thesis, University of Birmingham, 1998.
<http://ftp.cs.bham.ac.uk/pub/authors/T.Kovacs/index.html>.
- [6] Holland, J.H. (1975) *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975. Republished by the MIT press, 1992
- [7] Pier Luca Lanzi (1998) Generalization in Wilson's XCS. In AE. Eiben, T. Bäck, M. Shoenauer, and H.-P. Schwefel, editors, *Proceedings of the Fifth International Conference on Parallel Problem Solving From Nature -- PPSN V*, number 1498 in LNCS. Springer Verlag, 1998.
- [8] Pier Luca Lanzi, Wolfgang Stolzmann, and Stewart W. Wilson, editors (2000) *Learning Classifier Systems. From Foundations to Applications*, volume 1813 of *LNAI*. Springer-Verlag, Berlin.
- [9] Wilson, S.W. (1995) 'Classifier fitness based on accuracy', *Evolutionary Computation*, 3(2):149–175.

- [10] Wilson, S.W. (1998) Generalization in the XCS classifier system. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David E. Goldberg, Hitoshi Iba, and Rick Riolo, editors. *Genetic Programming 1998: Proceedings of the Third Annual Conference*. Morgan Kaufmann, pages 665-674.
- [11] Wilson, S.W. (2000a) 'Mining Oblique Data with XCS', Technical Report 2000028, University of Illinois at Urbana-Champaign, MI, USA.
- [12] Wilson, S.W. (2002) 'Compact Rulesets from XCSI', www.world.std/~sw/pubs.html, Presented at the Fourth International Workshop on Learning Classifier Systems (IWLCS-2001).
- [13] Database source: UCI Repository. Available: <http://www.ics.uci.edu/~mllearn/MLSummary.html>

Latent Semantic Indexing Based on Factor Analysis

Noriaki Kawamae

*NTT Information Sharing Platform Laboratories,
3-9-11, Midori-cho Musashino-shi Tokyo 180-8585 JAPAN
tel. +81-422-59-6197 fax. +81-422-59-3409
kawamae.noriaki@lab.ntt.co.jp*

Abstract. The main purpose of this paper is to propose a novel latent semantic indexing (LSI), statistical approach to simultaneously mapping documents and terms into a latent semantic space. This approach can index documents more effectively than the vector space model (VSM). Latent semantic indexing (LSI), which is based on singular value decomposition (SVD), and probabilistic latent semantic indexing (PLSI) have already been proposed to overcome problems in document indexing, but critical problems remain. In contrast to LSI and PLSI, our proposed method uses a more meaningful, robust statistical model based on factor analysis and information theory. As a result, this model can solve the remaining critical problems in LSI and PLSI. Experimental results with a test collection showed that our method is superior to LSI and PLSI from the viewpoints of information retrieval and classification. We also propose a new term weighting method based on entropy.

1. Introduction

With the advent of digital databases and communication networks, it is easy to obtain Web pages and electronic documents. It is thus necessary to develop information retrieval methods to simplify the process of finding relevant information. Document indexing is one important information retrieval method. Traditionally, documents have been indexed and labeled manually by humans. An important example is the idea of notional families in the work of H. P. Luhn [9]. The primary goal is to index documents with the same precision achieved by humans. To develop such document indexing methods, the following problems must be solved:

- *Ambivalence between terms
- *Calculation cost
- *Document indexing and keyword matching methods

Due to these problems, the retrieval performance of indexing systems is poor. Among previous works, latent semantic indexing (LSI)[4], based on singular value decomposition (SVD), and probabilistic latent semantic indexing (PLSI)[5] have been developed to overcome these problems, but unsolved problems remain.

Our primary goal in this paper is to present a novel statistical approach, to simultaneously mapping documents and terms into a latent semantic space. This approach can index documents better than by using individual indexing terms because the topics in a document are more closely related to the concepts described therein than to the index terms used in the document's description. Our method uses a more meaningful, robust statistical model that associates documents and terms with a latent semantic factor. This model is based on factor analysis and information theory and enables us to remove this noise and extract a better latent semantic space than other methods. As a result, documents can be indexed nearly as well as by humans. This is mainly because factor analysis is a better statistical model than SVD for capturing hidden factors.

2. Related work on document indexing and term selection

The vector space model (VSM) [11] is an approach to mapping documents into a space associated with the terms in the documents. The weighting of the terms in a document provides the document's coordinates in space, and the similarity between documents is measured in space.

Latent semantic analysis (LSA) [2] is an approach to mapping documents into a lower dimensional space than in VSM. This is accomplished by projecting term vectors into this space by using a model based on SVD. To date, several theoretical results or explanations have appeared, and these studies have provided a better understanding of LSI. However, many fundamental problems remain unresolved, as follows.

Inadequate use of statistical methods:

LSI uses SVD as a statistical model. Therefore, LSI cannot explain or extract the latent semantic factor as a hidden variable, because SVD is nothing more than the decomposition of the observed variable matrix [6].

Optimal decomposition dimension:

In general, dimensionality reduction is justified by the statistical significance of the latent semantic vectors as measured by the likelihood of the model based on SVD [3],[5]. The complexity of the model is not considered in terms of the dimension.

Term weighting method:

To date, many researchers on LSI have used *tf/idf* [11] or a similar method. This type of method, however, is not the best way to evaluate the usefulness of terms, because the weighting of low-frequency terms is underestimated, while that of high-frequency terms is overestimated.

3. Latent semantic extraction and term selection

Our method is a novel statistical approach to simultaneously mapping documents and terms into a latent semantic space, and it improves document retrieval performance. Our method consists of two components: (1) a novel term weighting method, and (2) our proposed model. The main idea in this approach is that a latent semantic factor is associated with each topic. A particular topic in a document is more related to the concepts described in the document than to the index terms used in the description of the document.

Therefore, this proposed indexing method enables us to retrieve documents based on similarities between concepts.

As a result, our proposed method evolves from keyword matching to concept matching. This allows us to retrieve documents even if they are not indexed by the terms in a query, because one document shares concepts with another document indexed by the given query. Therefore, the latent semantic space improves document retrieval performance.

3.1 Term-document matrix

Morphological analysis can be used to convert a document into a vector consisting of the terms occurring in it. The vector space model is a method of geometrically visualizing the relationships between documents. In this model, the relationships between terms and documents are represented as a term-document matrix, which contains the values of the index terms t occurring in each document d , properly weighted by other factors [4][12][8]. We denote m as the number of index terms in a collection of documents and n as the total number of documents. Formally, we let A denote a term-document matrix with n rows and m columns and let w_{ij} be an element (i, j) of A . Each w_{ij} is assigned a weight associated with the term-document pair (d_i, t_j) , where d_i ($1 \leq i \leq n$) represents the i -th document and t_j ($1 \leq j \leq m$) represents the j -th term. For example, using a *tf/idf* representation, we have $w_{ij} = tf(t_j - d_i)idf(t_j)$. $tf(t_j - d_i)$ means the frequency of t_j in document d_i and $idf(t_j)$ means the inverse of the document frequency. Thus, given the values of the w_{ij} , the term-document matrix represents the whole document collection. Therefore, each document can be expressed as a vector consisting of the weights of each term and mapped in a vector space:

$$A = \begin{pmatrix} w_{11} & \cdot & \cdot & \cdot & w_{1n} \\ \cdot & \cdot & & & \cdot \\ \cdot & & \cdot & & \cdot \\ \cdot & & & \cdot & \cdot \\ w_{m1} & \cdot & \cdot & \cdot & w_{mn} \end{pmatrix} \equiv (d_1 \cdot \cdot \cdot d_n) \equiv \begin{pmatrix} t_1 \\ \cdot \\ \cdot \\ \cdot \\ t_m \end{pmatrix}. \quad (1)$$

There are two methods of term weighting [8]: local and global. Among various weighting methods, those designated as L3, G2, and G3 are novel and have not been described previously [14].

3.1.1 Local weighting

This approach defines the weights w_{ij} in each document. Let P_{ij} ($1 \leq i \leq n$, $1 \leq j \leq m$) denote the occurrence probability of t_j in d_i . We ascribe significance to a term's occurrence, on the grounds that it represents a document's topics more than other factors do. Therefore, we base w_{ij} on the term occurrence probability P_{ij} in each document, and we define a local weighting L_{ij} as follows:

$$L_{ij} = P_{ij} * \log(1 + P_{ij}). \quad (2)$$

In contrast to other local weighting methods based on term frequency, this method reduces the effect of very high frequency terms by multiplying P_{ij} by a logarithm.

3.1.2 Global weighting

This approach defines the weights w_{ij} over all documents. Let P_{ij} ($1 \leq i \leq n$, $1 \leq j \leq m$) denote the relative occurrence probability of t_j in d_j . We ascribe significance to the probability of a term occurring over all documents on the grounds that a given term provides information for topic prediction and affects global performance. Entropy is a convenient metric for comparing the probability distribution of a term's occurrence. Therefore, we base w_{ij} on the entropy of the relative occurrence probability, and we define a global weighting G_j as follows:

$$G_j = 1 + \frac{1}{\log n} \sum_{i=1}^n p_{ij} \log p_{ij}. \quad (3)$$

Because this weighting is based on entropy, if a term occurs with equal probability in all documents, it is weighted as 1 (the maximum). To ensure that it does not exceed 1, this weighting is normalized through division by $\log n$, where n is the total number of documents.

3.2 Introduction of factor analysis to obtain latent semantic space

3.2.1 Our proposed model

The main theme in obtaining a latent semantic space is to capture the essential, meaningful semantic associations while reducing the amount of redundant, noisy information. Here, we propose a model to determine a document's coordinates in the latent semantic space. Our proposed is based on the hypothesis that terms in documents are generated from a latent semantic factor and a given document has a probability of belonging to some category. This model can be used to determine the coordinates of not only documents but also terms. The relationships between terms and latent semantic factors are similar to the relationship between encoded data and an information source in information theory.

We next describe the key idea in this model. We think that a latent semantic factor is an information source and is coded into a term in a document. Therefore, the relationship between terms and latent semantic factors can be defined as follows.

$P(t_i|l_k)$: Probability of latent semantic factor l_k generating term t_i ,

where $\sum_{i=1}^n P(t_i|l_i) = 1$. A term t_j can be generated from not only the latent semantic factor l_k but also another factor l_k .

The relationship between a document and a latent semantic factor can be defined in the following way:

$P(l_k|d_i)$: Probability of document d_i belonging to latent semantic factor l_k ,

where $\sum_{i=1}^n P(l_i|d_i) = 1$. Document d_i can belong to not only latent semantic factor l_k but also another factor l_k .

The weights w_{ij} , which represent the value of t_j in d_i , are used to combine these definitions into a joint probability model, resulting in the expression:

$$w_{ij} = P(t_j|d_i) = \sum_{k=1}^l P(t_j|l_k)P(l_k|d_i) + P_\varepsilon(t_j)P_\varepsilon(d_i), \quad (4)$$

where $P(t_j|d_i)$ denotes not the empirical occurrence but the statistical probability distribution of t_j in d_i , obtained by multiplying the local and global weightings; $P_\varepsilon(t_j)$ denotes the unique probability of t_j ; and $P_\varepsilon(d_i)$ denotes the unique probability of d_i . The reason for introducing these quantities is that $P(t_j|d_i)$ cannot actually be explained in terms of only the joint probability of $P(t_j|l_k)$ and $P(l_k|d_i)$. The difference between the statistical probability distribution $P(t_j|d_i)$ and the model distribution based on only the joint probability of $P(t_j|l_k)$ and $P(l_k|d_i)$ is considered as noise in information theory.

Notice that multiple documents can belong to some latent semantic factor at the same time. This is because the latent semantic factors are associated with the observed variances as t_j , d_i . Moreover, the latent semantic factors do not constrain documents to be orthogonal to each other.

Despite the similarity, the fundamental difference between the aspect model [5] in PLSI and this our proposed model is the unique probability. This difference affects the latent semantic factors obtained.

3.2.2 (P)LSI

Latent semantic indexing is the application of a particular mathematical technique, called singular value decomposition (SVD), to a term-document matrix. SVD (and hence LSI) is a least-squares method. The projection into a latent semantic space is chosen such that the representations from the original space are changed as little as possible when measured by the sum of the squares of the differences. As mentioned before, the object of LSI is to map documents into a vector space of the latent semantic factors, which has reduced dimensionality. The mapping is computed by decomposing the term-document matrix A . First, the SVD is computed by decomposing the term-document matrix into a product of three matrices. The singular value decomposition of A is defined as follows:

$$A = U\Sigma V^T.$$

The matrix U is a matrix of eigenvectors derived from a term-to-term correlation matrix given by AA^T . The matrix V is a matrix of eigenvectors derived from the transpose of the document-to-document correlation matrix given by $A^T A$. The matrix Σ is a diagonal matrix of singular values.

In the SVD, matrix A (dimensionality: $m \times n$) is decomposed to an $m \times k$ matrix A_k by a singular value selected from matrix Σ . The largest singular values in Σ are kept, along with their corresponding columns in U and V^T , while the remaining singular values become zero in this decomposition:

$$A = U\Sigma V^T \approx U_k \Sigma_k V_k^T = A_k.$$

This A_k is guaranteed to be the best approximation in the rank of A in the sense of an L_2 matrix norm. However, there is no clear cutoff point where the k largest values are well separated from the rest [3], in contrast to the proposed method, as described in section 2.

In PLSI, the empirical probability distribution $P(t_j|d_i)$, which is a component of A , can be written as $P(t_j|d_i)=P(t_j|z_k)P(z_k)P(d_i|z_k)$. The variance z_k may be the same as the variance l_k in the code model. Notice, however, that $P(t_j|z_k)$ is a component of U and $P(d_i|z_k)$ is a component of V^T . These matrices U and V are calculated principal component analysis (PCA). In other words, matrix U is a matrix of eigenvectors derived from AA^T and V is a matrix of eigenvectors derived from $A^T A$, as shown previously. Therefore, the variance z_k is not a latent semantic factor but is a collection of observed variables.

3.2.3 Factor analysis

To calculate the occurrence probability of terms in our proposed model, we introduce factor analysis, which is a statistical method that resolves an observed variance into a corresponding latent factor. Undoubtedly, SVD can also calculate a variable corresponding to the latent semantic factor obtained arithmetically by our method. The variable in SVD is a mixture of observed variables. In contrast, the variable in factor analysis is a latent factor. Therefore, it fits our proposed model mentioned earlier. In utilizing factor analysis, we define the observed variance as $P(t_j|d_i)$ and the latent factor as a concept.

3.3 Geometry of SLSI vs (P)LSI

Here, we show the exact difference between SLSI and (P)LSI from a geometrical viewpoint. A simple outline of the relationships between documents and terms in SLSI and (P)LSI is shown in Figures 1 and 2. In these figures the bold vectors represent term vectors and the squares represent the vector spaces. As discussed previously, (P)LSI maps documents into a reduced dimension of the VSM. This mapping is computed by decomposing the term-document matrix A . Therefore, each reduced axis is composed of term vectors. This causes these axes to be confined within the square of the VSM, as shown in Figure 1. These axes must be orthogonal to each other, according to the constraint of SVD, but this constraint is too strict to obtain a latent semantic space. Furthermore, it is strange to claim that these axes represent a latent semantic factor, because they are not composed of hidden variables but of observed variables. As a result, (P)LSI seems to be unfit for obtaining a latent semantic factor in view of not only statistics but also our intuition.

However, SLSI seems to be fit for obtaining a latent semantic factor from both viewpoints. This is because the axes, obtained by factor analysis, are free from any constraint or hidden variable factor, as shown in Figure 2. As stated before, factor analysis resolves an observed variance into a corresponding latent factor and does not constrain latent factors to be orthogonal to each other.

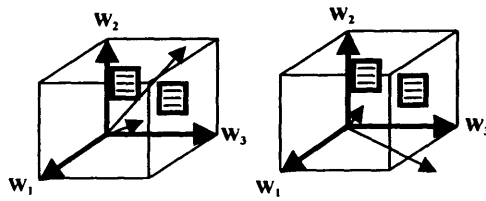


Figure 1: (P)LSI in VSM Figure 2: SLSI in VSM

3.4 Documents similarity

In information retrieval it is widely accepted that the cosine function between two document vectors in vector space measures not only the similarity of the terms in the documents but also the similarity between the documents. Matrix $A^T A$ contains the similarities between all pairs of documents. Therefore, the relationship between any two terms in a reduced space of dimension k can be obtained from the matrix given by

$$A^T A = (FT + UV)^T (FT + UV) = (FT)^T FT.$$

Note that the similarities between all documents are defined not by terms but by latent semantic factors. Similarly, the relationship between any two documents in a reduced space of dimension k based on SVD can be obtained from the matrix given by

$$\begin{aligned} A A^T &= (U_k \Sigma V_k^T)(U_k \Sigma V_k^T)^T \\ &= (U_k \Sigma V_k^T)(V_k \Sigma^T U_k^T) \\ &= U_k \Sigma_k \Sigma_k^T U_k^T \\ &= (U_k \Sigma_k)(U_k \Sigma_k)^T \end{aligned}$$

4. Document indexing in latent semantic space

4.1 Experimental design

Here, we assess the effectiveness of Our proposed method from three viewpoints: (1) term weighting methods, (2) Our proposed method vs. (P)LSI, and (3) statistical model selection based on SC.

Both (P)LSI and Our proposed method use the term-document matrix as a starting point and map documents into a latent semantic space. To compare the effectiveness of these methods, we evaluated the retrieval performance for latent semantic spaces based on each method. Or, in terms of obtaining a latent semantic space, we compared SVD and factor analysis. We evaluated term weighting in terms of the contribution to the latent semantic factors, and statistical model selection in terms of whether it can predict the optimal number of latent semantic factors.

For our experiments, we prepared 210 news articles to form a test collection. These articles covered seven topic categories: economics, entertainment, information technology, politics, society, sports, and world news. Each category included the same number of articles. We used these categories to judge retrieval and classification results in terms of average precision and recall rate.

First, we decomposed the articles into terms by using morphological analysis, which exchanges a sentence for its component terms. We used Chasen's approach [1] for this step. We used only terms that meet the condition that the part of speech is a noun or an unknown that is not registered in the dictionary of morphological analysis. This is because the terms, as used in queries, are limited to these parts of speech. We also omitted terms that did not appear in three or less different documents. Generally, terms with a size of 1 (for example, "a", &, @, etc.) were defined as stop words and omitted, and numbers were omitted as well. Single Japanese characters, however, do have meaning, so we did not treat these as stop words. The documents contain 7863 different terms, so a 7863×210 term-document matrix was generated.

4.2 Comparison of (P)LSI vs Our proposed method and term weighting methods

Table 1 compares similarity results among the documents based on different term weightings and latent semantic spaces. This comparison was done as follows. First, we calculated the documents' similarity matrix, as defined in section 3.2.4. Second, if the similarity of two documents was above a threshold, we judged these documents to be alike. Even if one of the documents did not include a term used in a query, we included both documents in the search results. Finally, we evaluated the search results in terms of the precision and recall rates, as shown in Table 1. In this experiment, we defined 0.7 as the threshold.

The normal vector space model consisted of 7863 term axes. The spaces decomposed by SVD or by factor analysis both consisted of 7 axes. In both cases, the precision and recall rate were highest for a decomposed space with this number of axes.

Table 1 indicates that L3 and L4 were the most effective term weighting methods for local weighting, while G1, G3, and G4 were the most effective for global weighting. As for the decomposed space, both SVD and factor analysis achieved higher values than the normal vector space model. These results show that the similarity between documents measured in a space decomposed by SVD or factor analysis reflects a fundamental relationship between topics in the documents. They do not, however, distinguish Our proposed method from (P)LSI. The term weighting methods used in this experiment are defined as follows.

Local weighting**L1: term occurrence probability**

Weights w_{ij} defined by occurrence in a document. $L1_{ij} = P_{ij} = \frac{C(t_{ij})}{\sum_{j=1}^{m_j} C(t_{ij})}$

L2: normalized term occurrence probability

Weights w_{ij} defined by a normalization of L1 based on a log function.

$$L2_{ij} = \log \left(1 + \frac{C(t_{ij})}{\sum_{j=1}^{m_j} C(t_{ij})} \right)$$

L3: normalized entropy of document

Weights d_j defined by a term's occurrence probability distribution in a document.

$$L3_i = 1 - \frac{1}{\log m_i} \sum_{j=1}^{m_i} p_{ij} \log p_{ij}$$

m_i : Number of different kinds of terms in document d_i .

p_{ij} : Frequency probability of term t_j in document d_i .

L4: proposed local weighting method**Global weighting****G1: normalized entropy of term occurrence over all documents**

Weights t_j defined by each term's occurrence over all documents. This method is similar to our proposed global weighting but differs in terms of P_{ij} .

$$G1_j = 1 + \frac{1}{\log n} \sum_{i=1}^n p_{ij} \log p_{ij}$$

P_{ij} : Occurrence probability of term t_{ij} over all documents.

G2: normalized entropy of term occurrence over all documents

Weights t_j defined by the occurrence proportion over all documents.

$$G2_j = -p_j \log p_j - (1 - p_j) \log(1 - p_j)$$

P_j : Occurrence probability of a document containing t_j over all documents.

G3: proposed global weighting method**G4: document frequency (=idf)**

Weights d_i defined by the inverse of the document frequency.

$$G4_j = 1 + \log \frac{n}{C(d_j)}$$

$C(t_j)$: Number of documents containing t_j .

G5: normalized entropy of relative term occurrence

Weights d_j defined by the relative probability distribution over all documents.

$$G5_j = 1 + \frac{1}{\log n} \sum_{i=1}^n p_{ij} \log p_{ij}$$

P_{ij} : Relative probability of t_j occurring in d_i .

Table 1.: Average precision and recall rates in for (P)LSI and Our proposed method

This table compares not only the term weighting methods but also the VSM, (P)LSI, and Our proposed method, in for each term weighting method. In For each same term weighting method, the upper top row is the VSM, the middle row is (P)LSI, and the lower bottom row is Our proposed method. Data listed as P/R. TW and P/R are defined below this table.

TW	Economics	Entertainment	IT Information	Politics	Sports	Society	World news	Average
L1G1	100/3.3	100/3.3	100/3.3	99.7/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	96.6/92.7	98.9/98.7	98.7/98.7	99.7/99.6	99.8/100	98.3/99.1	98.7/95.8	98.7/97.8
	96.3/92.4	98.9/98.9	98.6/99.1	100/99.8	99.8/99.6	98.1/99.1	98.7/95.6	98.6/97.8
L1G2	100/3.3	100/3.3	100/3.3	99.9/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	97.0/91.8	100/99.1	98.7/97.1	100/99.8	100/99.8	99.4/98.2	98.5/66.2	99.1/93.1
	96.7/90.0	100/99.1	98.8/98.0	100/99.8	100/98.9	99.4/96.7	98.6/62.2	99.1/92.1
L1G3	100/3.3	100/3.3	100/3.3	99.8/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	96.9/93.6	99.3/99.3	98.9/99.3	99.8/99.6	99.8/100	98.8/99.1	98.7/95.8	98.9/98.1
	96.5/92.9	99.4/99.8	98.7/99.3	100/99.8	99.8/100	98.8/99.1	98.7/95.6	98.8/98.1
L1G4	100/3.3	100/3.3	100/3.3	99.8/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	97.8/96.4	99.7/100	99.0/99.6	99.8/100	99.7/100	99.5/99.6	99.2/99.3	99.2/99.3
	97.5/96.4	99.5/100	98.8/99.6	100/100	99.5/99.6	99.5/99.3	99.1/99.3	99.1/99.2
L2G1	100/3.3	100/3.3	100/3.3	99.7/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	96.6/92.7	99.0/98.7	98.7/98.7	99.7/99.6	99.8/100	98.4/99.1	98.7/95.8	98.7/97.8
	96.3/92.4	99.0/98.9	98.6/99.1	100/99.8	99.8/99.6	98.2/99.1	98.7/95.6	98.6/97.8
L2G2	100/3.3	100/3.3	100/3.3	99.9/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	97.0/91.8	100/99.1	98.8/97.6	100/99.8	100/99.8	99.4/98.2	98.6/66.9	99.1/93.3
	96.8/90.2	100/99.1	98.8/98.2	100/99.8	100/98.9	99.4/96.7	98.6/62.2	99.1/92.2
L2G3	100/3.3	100/3.3	100/3.3	99.8/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	97.0/94.0	99.4/99.3	98.9/99.3	99.8/99.6	99.8/100	98.9/99.1	98.7/95.8	98.9/98.2
	96.5/93.1	99.4/99.8	98.7/99.3	100/99.8	99.8/100	98.8/99.1	98.7/95.6	98.8/98.1
L2G4	100/3.3	100/3.3	100/3.3	99.8/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	97.8/96.4	99.7/100	99.0/99.6	99.8/100	99.7/100	99.5/99.6	99.2/99.3	99.2/99.3
	97.6/96.4	99.5/100	98.8/99.6	100/100	99.5/100	99.5/99.3	99.2/99.3	99.1/99.2
L3G1	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
L3G2	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	99.9/99.8	100/100	99.9/100	100/100	100/100	100/100	100/97.1	100/99.6
	99.9/99.6	99.2/100	99.9/100	100/100	99.0/95.3	100/100	100/58.9	99.7/93.4
L3G3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
L3G4	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
	100/100	99.9/100	100/100	100/100	399.9/100	100/100	100/100	100/100
L4G1	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
L4G2	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	99.9/99.8	100/100	99.9/100	100/100	100/100	100/100	100/97.1	100/99.6
	99.9/99.6	99.2/100	99.9/100	100/100	99.0/95.3	100/100	100/58.9	99.7/93.4
L4G3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
L4G4	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.3	100/3.6	100/3.4
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100
	100/100	99.9/100	100/100	100/100	99.9/100	100/100	100/100	100/100

TW:term weighting, P:precision, R:recall

4.3 Evaluation of term selection

In this experiment, we evaluated the classification results by selecting terms in descending order of the value of $G1$ or $G3$ multiplied by $L3$ or $L4$. Figure 5 shows the precision of the classified documents. The documents were classified into seven groups based on the similarity matrix, which consisted of the values of selected terms. the value when the number of terms is under 2500. The results indicate that a latent semantic space based on factor analysis classifies documents better than one based on SVD, and our proposed weighting method ($L4 \cdot G3$) is also better than the other method. When we use all terms to classify documents, we cannot distinguish factor analysis from SVD in terms of precision. When we reduce the number of terms by using the value of global weighting, however, we see the superiority of factor analysis.

As for the weighting methods, the results indicate that it is effective to select terms based on the value with $L4$ as the local weighting method. We should emphasize that the latent semantic space based on the combination of factor analysis and $G3 \cdot L4$ can be used to classify documents with only the half number of terms normally required. The VSM, on the other hand, does not depend on the number of terms or the weighting method. these results show that our proposed weighting method is useful not only for indexing documents but also for selecting the minimum number of terms.

5. Conclusions

We have proposed a novel statistical approach to simultaneously mapping documents and terms into a latent semantic space. Our proposed method consists of two components: (1) a novel term weighting method, and (2) our proposed model.

Retrieval and classification experiments on a test collection indicated that Our proposed method is superior to (P)LSI. In other words, the axes in a latent semantic space obtained by our method are closer to the general concepts indexed by humans than any other method.

We thus conclude that our method is a useful document indexing method that not only solves the critical remaining problems in LSI and PLSI but also improves the retrieval performance.

REFERENCES

- [1] Chasen: <http://chasen.aistnara.ac.jp/index.html.ja>.
- [2] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R.: Indexing by latent semantics analysis. *Journal of the American Society for Information Science*, 1990.
- [3] Ding, C. H. Q.: A Dual Probabilistic Model for Latent Semantic Indexing in Information Retrieval and Filtering, in *Proceedings of the 22nd Annual Conference on Research and Development in Information Retrieval (ACM SIGIR)*, 1999.
- [4] Dumais, S.T.: Improving the retrieval of information from external sources, *Behavior Research Methods, Instruments and Computers*, 23(2), 229-236. 1991.
- [5] Hofmann, T.: Probabilistic latent semantic indexing, in *Proceedings of the 22nd Annual Conference on Research and Development in Information Retrieval (ACM SIGIR)*, 1999.
- [6] Kawamae, N., Aoki, T., Yasuda, H.: Information Retrieval Based on the Information Theory Model. Technical Report of IEICE, DE2001-57, 2001 (in Japanese).
- [7] Kawamae, N., Aoki, T., Yasuda, H.: Document Classification and Retrieval after Removing Word Noise. Technical Report of IEICE, NLC2001-48, 2001 (in Japanese).
- [8] Kita, K.: *Statistical Language Model*, The University of Tokyo Press, 1999.
- [9] Luhn, H. P.: The automatic derivation of information retrieval encodement from machine readable text. *Information Retrieval and Machine Translation*, 3(2), 1021-1028, 1961.
- [10] Schutze, H. and Pedersen, J.: A vector model for syntagmatic and paradigmatic relatedness, in *Proceedings of the 9th Annual Conference of the University of Waterloo Center for the New OED and Text Research*, 1993.
- [11] Salton, G. and McGill, M. J.: *Introduction to Modern Information Retrieval*, McGraw-Hill, 1983.
- [12] Salton, G. and Buckley, C.: Term-weighting approaches in automatic text retrieval, *Information Processing and Management*, 24(5) 513-523, 1988.
- [13] Saul, L., and Pereira, F.: Aggregate and mixed order Markov models for statistical language processing, in *Proceedings of 2nd International Conference on Empirical Methods in Natural Language Processing*, 1997.
- [14] Tohku, T.: *Information Retrieval and Language Process*. The University of Tokyo Press, 1999.

Document Oriented Modeling of Cellular Automata

Christian Veenhuis, Mario Köppen

Fraunhofer Institute for Production Systems and Design Technology

Department Pattern Recognition

Pascalstr. 8-9, 10587 Berlin, Germany

Email: {veenhuis|mario.koeppen}@ipk.fhg.de

Key Words: cellular automata, cellular automaton document,
modeling language, algorithm design

Abstract.

This paper proposes a document-oriented modeling concept for cellular automata (CA), which supports the simple and rapid design of a huge variety of cellular automata. This modeling concept is realized as a domain-specific modeling language derived from XML (eXtensible Markup Language). XML is in general considered as the future for internet documents and data exchange. The main concept behind XML is to separate the content of a document from its layout (its appearance). The presented modeling concept uses a document for describing a whole cellular automaton. Like the content of a document is separated from its layout, the abstract cellular automaton is separated from a concrete implementation and programming language. Everyone can create and use XSL(T) stylesheets for translating cellular-automaton-documents into ready to use source-code (covering the adequate cellular-automaton-functionality) as well as for documentation and exchange of the realised CA.

1 Introduction

A cellular automaton (CA) is a system consisting of units called cells whose discrete states change over time. These cells are arranged by a specific type of lattice, as depicted in figure 1. There is a variety of possible lattices. Lattice type (a) defines a hexagonal grid, where every cell has six neighbors. Lattice type (b) defines a one dimensional square lattice with each cell having two neighbors. The square lattices (c) and (d) can either have a five-neighborhood (*von Neumann neighborhood*) or a nine-neighborhood (*Moore neighborhood*).

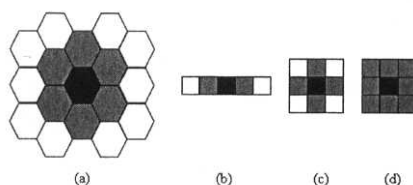


Figure 1: Different lattices for cellular automata

A neighborhood is assigned to each cell, according to the given lattice type. Furthermore, every cell possesses a rule (or rule set) which determines the state of this cell in dependence on its neighbors. According to [9] and [3] a one dimensional CA can be defined by equation 1, whereby r defines the range for the neighbors. In figure 1 the lattice type (b) uses $r = 1$.

$$a_i^{(t+1)} = \Phi(a_{i-r}^{(t)}, \dots, a_i^{(t)}, \dots, a_{i+r}^{(t)}) \quad (1)$$

The state of the cell for the next generation $t + 1$ can be computed by a function Φ . For computing the next state, the function Φ uses the appropriate neighbors.

Analogous to equation 1 a two dimensional CA with a five-neighbor square can be defined as in equation 2. CAs with any number of neighbors and even with multiple dimensions can be defined similarly to equation 2.

$$a_{i,j}^{(t+1)} = \Phi(a_{i,j}^{(t)}, a_{i,j+1}^{(t)}, a_{i,j-1}^{(t)}, a_{i+1,j}^{(t)}, a_{i-1,j}^{(t)}) \quad (2)$$

Cellular automata are used in various fields, for example, biology, chemistry, and physics to simulate (non-linear) phenomena like computational epidemics, diffusion of particles or the flow of fluids. CAs are also used for computational, cryptographic, and image processing tasks.

In the past, some activities were directed at the implementation of software libraries covering CA functionality. Furthermore, there already exist languages like *cellang* for the development of CAs [2]. These languages are specifically designed for the domain of CAs and are often interpreted. A lot of software systems are available to simulate and explore CAs. Unfortunately, the created and defined CAs can't be exchanged between these software systems. If one wants to use a created CA with another simulator, the CA has to be re-programmed or re-defined. Furthermore, if one has written a CA in a programming language (e.g. C, C++, Java) and needs to use this CA on another platform or in another programming language, the CA has to be re-programmed, too.

When developing or applying cellular automata it is useful to handle a specific CA on a more abstract level than the level of programming languages or function-calls. To overcome the mentioned problems and to revive efforts in exchange and standardization, a concept for a cellular-automaton-document is proposed, which supports the modeling, description and documentation of CAs at once. The cellular-automaton-document (CAD) should be system-independent and separated from an explicit implementation. It should also be easy to understand and to learn.

Looking for a suitable concept, different approaches have been studied. Among them are the Extensible Markup Language (XML) [1] [8] and Evolutionary Algorithm Modeling Language (EAML) [4] [6], whereby EAML is derived from XML. Both languages are not traditional programmer-languages that specify a computation flow. They are developed to specify and characterize elements of an environment. These languages abstract several elements by adapted constructs and support their hierarchical order within a document. As an example EAML is used to model Evolutionary Algorithms (EA) like Genetic Algorithms, Genetic Programming, Evolutionary Strategies, and so on. Every EA is built up by a hierarchical order of elements, which represent the different (genetic) operators and methods. This way a semantic model is constructed: the kind of order and the nesting of the elements represent the specific EA.

The authors propose a concept for a Cellular Automata Modeling Language (CAML) [5], which is derived from XML and works similar to EAML: Several elements are ordered and

nested to describe any set of rules and global properties on an abstract level. Finally, the proposed element-hierarchy realizes a complete CA. Using XML-notation documentation and exchange of a CA could be nice. Because CAML is derived from XML, it 'inherits' the platform independency, readability (for humans (text-based) and software (structured)), and the more general and abstract handling because it is independent of the implementation (programming language). Therefore a CAML document can be exchanged between all software simulators, which possess a CAML interpreter. Furthermore, a CA described by a XML notation can be translated into various source-codes (C,C++,Java etc.) by using stylesheets (see figure 6). For this, XSL(T) (XML Stylesheet Language (Transformations)) can be used. By using stylesheets, one does not need to be familiar with compiler technology to translate CAs into source-code. In addition, CAs modeled with CAML can be translated or transformed into every text-based format. Thus, documentations could be generated automatically by using an appropriate stylesheet. It might even be possible to translate CAML documents into hardware description languages like VHDL (but this point has to be examined in the future).

Section 2 presents the current state of CAML. To find out whether CAML is suitable for practical applications or not, a CAML-interpreter was implemented concomitantly and will be introduced briefly in section 3. Finally, section 4 gives an outlook on to further developments. The authors would like to motivate further contributions for the extension of CAML as well as for standardization and exchange within the CA-community.

2 Derivation of the Cellular-Automaton-Document

2.1 Components of a cellular automaton

To be able to model a CA, its specific components have to be recalled. In general, a CA mainly comprises the following components:

- A *lattice*, which arranges all cells.
- At least one *state* with a valid range of values defined by an interval $[k_{min}, k_{max}]$.
- A range for the local *neighborhood* (the r in equation 1).
- A set of at least one *rule* (the Φ in equation 1 and 2).
- A set of logical functions for the use with the neighbor cells as *conditions* for rules.
- *State-readers*, which can read the state of neighbors and the current cell.
- A *state-writer*, which can set the current cell to a new state. Only the current cell should be manipulable. If we could change also the neighbors, we wouldn't have a CA.

Although upcoming components might not be predictable, hence, the modeling concept for a Cellular-Automaton-Document (CAD), that describes CAs, has to consider the huge variability of behavioural and structural design of a CA. The different components of a CA, rules and computations, could be handled as building blocks. So, it will be possible to fiddle about with those building blocks, being simply called *elements*. Finally, a complete CA could be considered as a nested arrangement of elements. For user convenience the most common computations should be pre-defined.

2.2 Notation and basic concepts of CAML

The notation and the basic concept of CAML follows the XML-standard [1]. In general, the structure of CAML consists of several elements, and an ordered collection of such elements can be called a CA-description. Together with some administrative XML elements, such a CA-description is stored in an ASCII-file. This ASCII-file is called CAML document and can be created with a simple text-editor or even with every XML editor by using the Document Type Definition (DTD) of CAML. A DTD is a XML description (grammar) for the structure of a document (language), which is derived from XML [1].

The different elements are applied by the following general notation:

```
< NameOfElement
  Attribute[i].name = "Attribute[i].value"
>
  content (...further nested elements...)
</ NameOfElement>
```

Each element is introduced by a so-called start-tag (<NameOfElement>) and finished with an end-tag (</NameOfElement>). Enclosed in these both tags the element-content is written. Depending of the element-type an attribute-list containing further parameters can be specified. It has to be considered that attributes are able to carry default-values. Therefore, they haven't to be specified explicitly. Some elements might also contain further elements. Therefore, it is spoken of nested elements or an element-hierarchy.

The definition-notation for the element-types and their attributes were derived from XML [1] and will not be explained further at this point.

Each CA-functionality, like conditions, actions or computations, required for the description of a CA, has to be abstracted by the mentioned elements. So, up to now, every CA-functionality might be handled as complete parameterized building blocks. To give an example, an expected one dimensional pattern can be described by the following notation:

```
< Pattern dimx="3" centerx="1" > 1 0 1 </ Pattern>
```

The element is named after its meaning and comprises two attributes. With the dimx-attribute the length of the pattern can be determined. The centerx-attribute defines the position of the current considered cell within the pattern. As mentioned before, the attribute-values might be changed in a specific CA-description. Another example that should be provided is a *Neighbor*-element that provides the reading of the neighbor cells (now for a two dimensional CA):

```
< Neighbor x="-1" y="-1"></ Neighbor>
```

and shortened according to the XML-standard:

```
< Neighbor x="-1" y="-1" />
```

This element possesses two attributes, which contain the offsets to the neighbor cell. This way any number of neighbors with any range can be realized (the r in equation 1).

By using the presented notation all components of a CA can be modeled, no matter whether they are rules, state-readers, state-writers, conditions, and so on.

2.3 The element-hierarchy

A complete CA-description consists of diverse elements arranged in a nested order. So, these elements form a hierarchy, which is mentioned as element-hierarchy. This element-hierarchy also represents the affiliation of an element, whereby each element of a lower level is contained in an element of the higher level. Currently, on the top-level there are the following elements permitted:

- CA Covers the complete description of one CA
- Rule Describes a (single) rule

As suborder-elements of the CA-element, up to now, the following elements were provided. Moreover, some of these elements contain further elements as content.

- Conditions EVER, OR , AND , NOT , Pattern , GT , LT , EQ , UNEQ
- Operators Break , Literal , Min , Max , Add , Sub , Mul , Div , Sum
- State-readers Neighbor , Current
- State-writers Current
- Structure Cond , Action , Otherwise

The number of elements is not restricted. From a CA-element hierarchically downwards all components of a particular CA are specified.

Figure 5 depicts the Document Type Definition (DTD) of the current state of CAML. This DTD defines the grammar of valid CAML documents. For this it defines the affiliations of the elements and their attributes.

In figure 2 a simple example of a valid CAML document is given. There, Conway's *Game of Life* is described with CAML. This is a very old application of CAs, but it's well-known and therefore a good example for a CAML document. The description of a CA is realized by several sub-trees with the root being a *Rule*-element. A single sub-tree contains e.g. a condition (for activation of a rule) or an action which computes and sets the new state for the current cell. The example in figure 2 contains two rules for describing the rules of the *Game of Life*:

1. If a cell is dead, it will become alive if it has exactly 3 neighbors which are alive
2. If a cell is alive, it will keep alive if it has 2 or 3 neighbors which are alive (otherwise it will die)

Every *Rule*-element can contain any number of *Cond*-, *Action*- and *Otherwise*-elements. Each of these elements can contain several sub-trees of theoretically infinite depth for describing the rules in all complexity. This way it is possible to design a wide variety of different CA-descriptions.

Equation 3 shows the definition of a CA described with CAML. A rule (R_n) consists of conditions (C_n), actions (A_n) and otherwise (O_n) realized by the appropriate CAML elements. A whole CA Φ contains several rules (R_n).

$$\begin{aligned}
S_k &:= k^{th} \text{ sub-tree} \\
C &= \{S_1, \dots, S_{n_{trees}}\} \\
A &= \{S_1, \dots, S_{n_{trees}}\} \\
O &= \{S_1, \dots, S_{n_{trees}}\} \\
R &= \{C_1, \dots, C_{n_{conditions}}\} \cup \{A_1, \dots, A_{n_{actions}}\} \cup \{O_1, \dots, O_{n_{otherwise}}\} \\
\Phi &= \{R_1, \dots, R_{n_{rules}}\}
\end{aligned} \tag{3}$$

These rules are executed in the order of their position within the CA-description. For every rule the activity has to be computed to decide whether the action or the otherwise part must be executed. The activity for a given rule R_n can be computed like in equation 4: At least one *Cond*-element (C_i) of the *Rule*-element (R_n) should evaluate to 1. For this, each of the n_{trees} sub-trees of this *Cond*-element must evaluate to 1, too.

$$\exists C_i \in R_n \left(\bigwedge_{k=1}^{n_{trees}} S_k \in C_i \right) \tag{4}$$

If a rule R_n of the CA fulfills equation 4, all *Action*-elements (otherwise all *Otherwise*-elements) of this rule are executed.

In figure 2 the conditions are defined by combining relational and logical operators with arithmetic functions. This resembles to syntax-trees of programming languages, where expressions are managed by tree-structures. Defining conditions this way can be very flexible but also bloating. The obtained CAML-document could be somewhat unreadable for human beings. Therefore, one is able to define conditions directly by specifying patterns like in figure 3. This way one has a more concise style than by using arithmetical computations. Figure 3 describes the Sierpinsky Triangle as shown in figure 4. Both styles (like in the figures 2 and 3) can be combined to describe as much CAs as possible.

3 Realisation

Concomitant with the CAML-concept, an interpreter was implemented for developing and testing of CAML descriptions. This was necessary in order to prove whether the proposed document-oriented model is meeting practical expenses.

The main concept behind XML is to separate the content of a document from its layout (its appearance). The layout is defined by a separate stylesheet language (CSS or XSL(T)). Like the content of a document is separated from its layout, the abstract cellular automaton is separated from a concrete implementation and programming language by a CAML document.

If one wants to use the CA behind the CAML document, one needs to combine the CAML document with an appropriate stylesheet. With this combination, the CAML document will be translated into a 'layout' defined by the stylesheet. This 'layout' is the concrete implementation of the CA. Figure 6 shows the whole environment for CAML documents: The CAML document represents one side of the input to a XSL processor. A XSL processor is a software tool which translates XML documents with respect to a XSL stylesheet. Such a XSL processor is mainly used within web servers to provide XML on web sites for producing HTML

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE CAML SYSTEM "caml.dtd">

<!--
  Game of Life: Assumes, that a dead cell has a 0 and a cell alive has a 1 as state.
-->

<CA project="gol" dimensions="2" min="0" max="1">

  <!-- ##### RULE 1 ##### -->

  <Rule>
    <!-- A dead cell becomes alive with exactly 3 living neighbors -->

    <Cond>
      <EQ>
        <Current/>
        <Literal value="0"/>
      </EQ>
      <EQ>
        <Sum>
          <Neighbor x="-1" y="-1"/> <Neighbor x="0" y="-1"/> <Neighbor x="1" y="-1"/>
          <Neighbor x="-1" y="0"/>
          <Neighbor x="1" y="0"/>
          <Neighbor x="-1" y="1"/> <Neighbor x="0" y="1"/> <Neighbor x="1" y="1"/>
        </Sum>
        <Literal value="3"/>
      </EQ>
    </Cond>

    <Action> <Current> <Literal value="1"/> </Current> </Action>

  </Rule>

  <!-- ##### RULE 2 ##### -->

  <Rule>
    <!-- A living cell keeps alive with 2 or 3 living neighbors, otherwise it dies -->

    <Cond>
      <EQ>
        <Current/>
        <Literal value="1"/>
      </EQ>
      <OR>
        <EQ>
          <Sum>
            <Neighbor x="-1" y="-1"/> <Neighbor x="0" y="-1"/> <Neighbor x="1" y="-1"/>
            <Neighbor x="-1" y="0"/>
            <Neighbor x="1" y="0"/>
            <Neighbor x="-1" y="1"/> <Neighbor x="0" y="1"/> <Neighbor x="1" y="1"/>
          </Sum>
          <Literal value="2"/>
        </EQ>
        <EQ>
          <Sum>
            <Neighbor x="-1" y="-1"/> <Neighbor x="0" y="-1"/> <Neighbor x="1" y="-1"/>
            <Neighbor x="-1" y="0"/>
            <Neighbor x="1" y="0"/>
            <Neighbor x="-1" y="1"/> <Neighbor x="0" y="1"/> <Neighbor x="1" y="1"/>
          </Sum>
          <Literal value="3"/>
        </EQ>
      </OR>
    </Cond>

    <Action> <Current> <Literal value="1"/> </Current> </Action>

    <Otherwise> <Current> <Literal value="0"/> </Current> </Otherwise>

  </Rule>

</CA>

```

Figure 2: Conway's *Game of Life* written in CAML

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE CAML SYSTEM "caml.dtd">

<!-- Sierpinsky Triangle (cell-values: 0 = dead, 1 = alive) -->

<CA project="sierpinsky" dimensions="1" min="0" max="1">

  <Rule>

    <Cond>
      <OR>
        <Pattern dimx="3" centerx="1" > 0 0 0 </Pattern>
        <Pattern dimx="3" centerx="1" > 1 1 1 </Pattern>
      </OR>
    </Cond>

    <Action>    <Current> <Literal value="0"/> </Current> </Action>

    <Otherwise> <Current> <Literal value="1"/> </Current> </Otherwise>

  </Rule>

</CA>

```

Figure 3: The Sierpinsky Triangle written in CAML

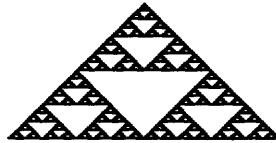


Figure 4: Sierpinsky Triangle

pages. Additionally, a XSL processor can be used autonomously. Therefore, we can use a XSL processor as 'compiler' to translate the CAML document e.g. into programming languages like C. The other input side of this XSL processor has to be the XSL stylesheet. After execution the XSL processor creates the output file, which will be the implementation of the CA. This output file could be compiled with a main-program or be integrated into another software project. This way, even people not familiar with compiler technology can realize a translation of CAML documents simply by defining XSL stylesheets. They can use every XSL processor as 'compiler' and every XML editor as development environment.

Furthermore, a CAML document can be interpreted by software systems. A CAML document is a general representation of a CA. CAML can be used as representation for the exchange of CAs as well as for the description and development of CAs. One representation allows a varied usage of the CA.

4 Conclusion and future work

Within this paper a document-oriented model following the XML-notation (Extensible Markup Language) was presented that enables the modeling of cellular automata. In this way it is possible to handle CAs with their components at the abstraction-level of building blocks. The document-oriented model applicable by CAML (Cellular Automata Modeling Language) enables the modeling as well as the documentation of a CA simultaneously and ensures the interchange of CAs modeled once.

```
<ENTITY % bool "true | false" >

<!ENTITY % dispatch_operators
    "(Break | Literal | Min | Max | Add | Sub | Mul | Div | Sum | Neighbor | Current)" >

<!ENTITY % dispatch_conds
    "(EVER | OR | AND | NOT | Pattern | GT | LT | EQ | UNEQ)" >

<!ELEMENT CA (Rule+)* >
<!--CA project CDATA "CA" -->
<!--CA dimensions CDATA "2" -->
<!--CA min CDATA "1.7e-307" -->
<!--CA max CDATA "1.7e+307" -->
<!--CA discrete (%bool;) "true" -->

<!ELEMENT Rule (Cond+ | Action+ | Otherwise+)* >

<!ELEMENT Cond (%dispatch_conds;)* >

<!--EVER EMPTY -->
<!--OR (%dispatch_conds;)+ -->
<!--AND (%dispatch_conds;)+ -->
<!--NOT (%dispatch_conds;)+ -->
<!--Pattern (#PCDATA) -->
<!--Pattern dimx CDATA "3" -->
<!--Pattern dimy CDATA "0" -->
<!--Pattern dimz CDATA "0" -->
<!--Pattern centerx CDATA "1" -->
<!--Pattern centery CDATA "0" -->
<!--Pattern centerz CDATA "0" -->
<!--GT (%dispatch_operators;, %dispatch_operators;) -->
<!--LT (%dispatch_operators;, %dispatch_operators;) -->
<!--EQ (%dispatch_operators;, %dispatch_operators;) -->
<!--UNEQ (%dispatch_operators;, %dispatch_operators;) -->

<!ELEMENT Action (%dispatch_operators;)+ >

<!--Otherwise (%dispatch_operators;)+ -->

<!--Break EMPTY -->
<!--Literal EMPTY -->
<!--Literal value CDATA #REQUIRED -->
<!--Min (%dispatch_operators;)+ -->
<!--Max (%dispatch_operators;)+ -->
<!--Add (%dispatch_operators;)+ -->
<!--Sub (%dispatch_operators;)+ -->
<!--Mul (%dispatch_operators;)+ -->
<!--Div (%dispatch_operators;)+ -->
<!--Sum (%dispatch_operators;)+ -->

<!--Neighbor EMPTY -->
<!--Neighbor x CDATA "0" -->
<!--Neighbor y CDATA "0" -->
<!--Neighbor z CDATA "0" -->

<!--Current (%dispatch_operators;)? -->
```

Figure 5: The Document Type Definition of CAML

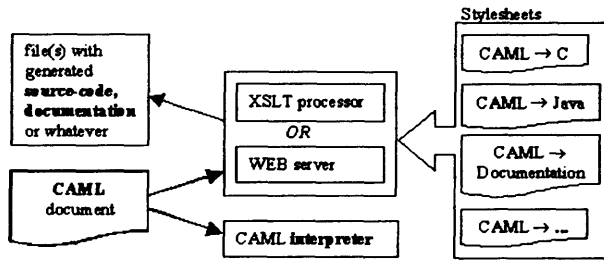


Figure 6: Transforming CAML documents into source code

Up to now, well-known CAs were modeled with CAML. To validate the designed CA documents, a CAML-interpreter was implemented. However, there is still some work to do. For example, it seems to be useful to model movable cells, e.g. for the simulation of particles. For this, every position within the lattice must be able to hold several cells at one time step. Up to now, every cell has only one state. Multiple states per cell could be a nice feature. The possibility of heterogeneous cells as well, where every cell can have different rules (or rule sets). At present CAML supports only rectangular lattices (e.g. type b, c and d in figure 1). The extension to further lattices would be a nice feature, too. Furthermore, several rule sets, which can be activated alternately, could be an enrichment. And, of course, the definition of mathematical formulas as rules should be possible. For this, it should be examined whether other XML based modeling languages like MathML [7] could be integrated into CAML.

Last but not least, the authors would like to motivate discussions and further contributions for the extension of CAML as well as for standardization and exchange within the CA-community.

5 Acknowledgement

The authors would like to thank Stephanie Wenzel for proof-reading this paper.

References

- [1] Henning Behme and Stefan Mintert, *XML in der Praxis: Professionelles Web-Publishing mit der Extensible Markup Language*, Addison-Wesley, Bonn, Deutschland, 1998
- [2] The Cellular Automata Simulation System, URL: <http://www.cs.runet.edu/dana/ca/cellular.html>, Dec 22, 2001
- [3] N.H. Packard and S. Wolfram, *Two-Dimensional Cellular Automata*, Journal of Statistical Physics, March 1985
- [4] C. Veenhuis, K. Franke, M. Köppen, *A Semantic Model for Evolutionary Computation*, Proc. 6th International Conference on Soft Computing (IIZUKA2000), Iizuka, Japan, 2000
- [5] C. Veenhuis, *Cellular Automata Modeling Language*, URL: <http://vision.fhg.de/veenhuis/CAML>, August 07, 2002
- [6] C. Veenhuis, K. Franke, M. Köppen, *Evolutionary Algorithm Modelling Language*, URL: <http://vision.fhg.de/veenhuis/EAML>, February 12, 2002
- [7] W3C (World Wide Web Consortium), *Mathematical Markup Language (MathML) Version 2.0*, URL: <http://www.w3.org/TR/MathML2>, Feb 24, 2002
- [8] W3C (World Wide Web Consortium), *Extensible Markup Language (XML) 1.0 (Second Edition)*, URL: <http://www.w3.org/TR/2000/REC-xml-20001006>, Feb 24, 2002
- [9] S. Wolfram, *Cellular automata*, Los Alamos Science, Fall 1983

Section 6

Support Vector Machines and Kernel Methods

This page intentionally left blank

An Empirical Comparison of Kernel Selection for Support Vector Machines

A B M Shawkat Ali and Ajith Abraham*

School of Computing and Information Technology, Monash University, Victoria 3842, Australia.

Email: Shawkat.Ali@infotech.monash.edu.au

**Department of Computer Science, Oklahoma State University, 700 N Greenwood Avenue, Tulsa, OK 74106, USA, Email: ajith.abraham@ieee.org*

Abstract. Support Vector Machine (SVM) has gained much attention as an efficient pattern recognition tool primarily between two classes problems by finding a decision surface determined by certain points of the learning set, termed Support Vector (SV). In this paper, we examine how to discriminate SVM for two class classification problems with different kernel settings. We also compare SVM with other three popular learning algorithms, namely Navie Bayes, C4.5 and neural network in terms of accuracy and computational complexity. Our studies reveal that SVM is the best choice for classification and SVM polynomial kernel is the best choice when compared to others.

1 Introduction

Support Vector Machines [1-5] are powerful tools for providing solutions to classification and function approximation problems. Classification could be visualized as a linear or nonlinear predictive modeling. According to Vapnik, traditional learning techniques for pattern recognition are based on the minimization of the empirical risk, in an attempt to optimize the performance of the learning set. On the other hand, SVMs minimize the structural risk by taking into account of the probability of misclassifying yet to be seen patterns for a fixed but unknown probability distribution of the data. This new induction principle is equivalent to minimization of an upper bound on the generalization error. However, SVMs use a linear separating hyperplane to create a classifier, yet it is not easy to linearly separate some problems in the original input space. SVMs can easily non-linearly transform the original input space into a high dimensional feature space. In this feature space, it is trivial to find a linear optimal separating hyperplane. This hyperplane is optimal in the sense of being a maximal margin classifier with respect to the learning set.

Ali [6] observed that SVM is giving the maximum efficiency for different datasets for linear classification; but it is computationally expensive when compared to other machine learning techniques. Amari and Wu [7] proposed a kernel function modification to improve the SVM performance. They only considered the Gaussian radial basis function kernels, leaving aside linear, polynomial, and sigmoidal kernels. Scholkopf *et al* [8] formulated an experimental comparison of SVMs with a classical approach, where the centers are determined by *k*-means clustering, and the weights are computed using error backpropagation. They considered three kernels, namely, a classical Radial Basis Function (RBF), Gaussian and a hybrid system with the centers determined by the Support Vector (SV) method and the weights trained by backpropagation algorithm. Scholkopf *et al* [9] discussed how the metric governing the intrinsic geometry of the mapped surface can be computed in terms of the kernel, using the example of the class of inhomogeneous polynomial kernels, which are often used in SV pattern recognition. Scholkopf *et al* [10] also explained a new criterion taking the distribution of Gram matrix of the data. According

to Joachims [11], SVM is the best choice for text classification by choosing the polynomial and radial basis kernels. Which is the best kernel for a particular problem? This is an old question, but with the inclusion of many mappings within one framework, it is easier to formulate a comparative study. In this paper, we attempt to investigate the best choice among SVM kernels namely linear, polynomial, and radial basis function kernels and a comparative study with three popular machine-learning algorithms.

We have organized the paper as follows. In Section 2, we present three popular classifiers and commonly used SVM kernels followed by some experimentation results and comparison of the different approaches in Section 3. Some conclusions and future directions are also provided towards the end.

2 Classifier Descriptions

2.1 Naive Bayes

The Naive Bayes algorithm computes a discriminant function for each of n possible classes. Let E be an example vector, with a features $\{X_1, \dots, X_s\}$, and $B_i(E)$ the discriminant function corresponding to the i th class. The chosen class, C_k , is the one for which

$$B_k(E) > B_i(E) \forall_i \neq k. \quad (1)$$

The discriminant function $B_i(E)$ is defined as

$$B_i(E) = \Pr(\text{Class} = C_i) \prod_{j=1}^a \Pr(X_j = v_j \mid \text{Class} = C_i) \quad (2)$$

where v_j is the value of the feature X_j in example E . The classification rule might be changed to reflect some desired operating conditions: in a two class problem the rule might be changed such that if one discriminant were above a given threshold, then that class would be assigned, regardless of the value of the other discriminant [12].

2.2 C4.5

A decision tree uses a top down divide-and-conquer strategy. It attacks a complex problem by dividing it into simpler problems and recursively applying the same strategy to the sub-problems. Solutions of sub-problems can be combined to yield a solution of the complex problem. This is the basic idea behind the well-known decision tree based algorithms, such as C4.5, CART etc [13]. C4.5 [14] decision tree is an information source that conveys a message concerning the classification of an object and letting p and n denote objects of different classes and the expected information content of the message is

$$I(p, n) = -\frac{p}{p+n} \log\left(\frac{p}{p+n}\right) - \frac{n}{p+n} \log\left(\frac{n}{p+n}\right) \quad (3)$$

where the probability of each possible message is computed from the learning set. If attribute A is selected, which has k different values, the resulting tree has expected information content of

$$E(A, p, n) = \sum_{i=1}^k \frac{p_i + n_i}{p + n} I(p_i, n_i) \quad (4)$$

where p_i and n_i are the number of instances from each class in the subtree associated with the partition I based on the values of the attribute A . The information gain achieved is

$$\text{gain}(A, p, n) = I(p, n) - E(A, p, n) \quad (5)$$

The corresponding heuristic selects the attribute that results in the maximum gain for that step.

2.3 Neural Network (NN)

Backpropagation (BP) is an algorithm for modifying the weights of a Multilayered Perceptron based on incremental gradient descent of mean-square error. BP aims at minimizing the squared error cost function over a training set.

$$E = \sum_{i=1}^{N_{\text{train}}} \left\{ \frac{1}{2} \sum_{j=1}^J (d_{ij} - y_{ij})^2 \right\} \quad (6)$$

where i indexes each pattern in the training set and j indexes each output variable (N_{train} patterns in the training set; each pattern has J outputs); d_{ij} is the desired value of output j as given by example pattern i , and y_{ij} is the actual output value from the model[15].

2.4 Support Vector Machine

Support Vector Machines (SVMs) [1,4,16-20] combine several techniques from statistics, machine learning and neural networks. SVM perform structural risk minimization. They create a classifier with minimized VC (Vapnik and Chervonenkis) dimension. If the VC Dimension is low, the expected probability of error is low as well, which means good generalization. SVM has the common capability to separate the classes in the linear way. However, SVM also has another specialty that it is using a linear separating hyperplane to create a classifier, yet some problems can't be linearly separated in the original input space. Then SVM uses one of the most important ingredients called kernels, i.e., the concept of transforming linear algorithms into nonlinear ones via a map into feature spaces. Figures 1 and 2 illustrate two categories of data using Y+ and Y- symbols.

2.4.1 Linear SVM

We consider N training data points $\{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ where $x_i \in \mathbb{R}^d$ and $y_i \in \{\pm 1\}$. We would like to explain a linear separating hyperplane classifier:

$$f(x) = \text{sgn}(w \cdot x - b) \quad (7)$$

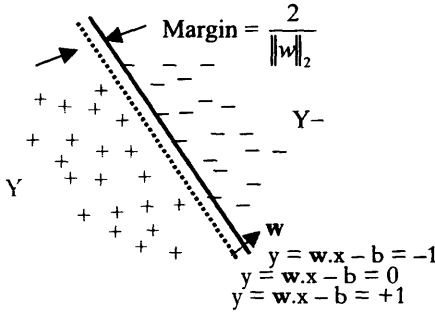


Figure 1: The linearly separable case.

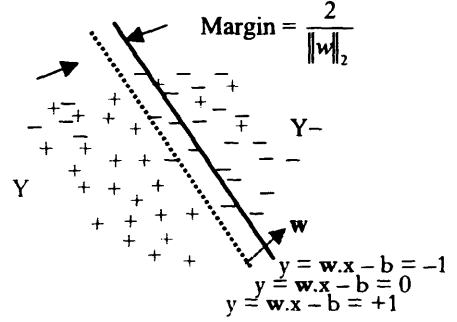


Figure 2: The linearly inseparable case.

Furthermore, we want this hyperplane to have the maximum separating margin with respect to the two classes. Specifically, we want to find this hyperplane $HP : y = w \cdot x - b = 0$ and two hyperplanes parallel to it and with equal distances to it,

$$HP_1 : y = w \cdot x - b = +1 \quad \text{and} \quad HP_2 : y = w \cdot x - b = -1 \quad (8)$$

with the condition that there are no data points between HP_1 and HP_2 , and the distance between HP_1 and HP_2 is maximized.

For any separating plane HP and the corresponding HP_1 and HP_2 , we can always normalize the coefficients vector w so that HP_1 will be $y = w \cdot x - b = +1$, and HP_2 will be $y = w \cdot x - b = -1$.

Our aim is to maximize the distance between HP_1 and HP_2 . So there will be some positive examples on HP_1 and some negative examples on HP_2 . These examples are called support vectors because only they participate in the definition of the separating hyperplane, and other examples can be removed and/or moved around as long as they don't cross the planes HP_1 and HP_2 .

Recall that the 2-D, the distance from a point (x_0, y_0) to a line $Ax + Bx + C = 0$ is $\frac{|Ax_0 + By_0 + C|}{\sqrt{A^2 + B^2}}$. Similarly, the distance of a point on HP_1 to $HP : w \cdot x - b = 0$ is $\frac{|w \cdot x - b|}{\|w\|} = \frac{1}{\|w\|}$, and the distance between HP_1 and HP_2 is $\frac{2}{\|w\|}$. So, in order to maximize the distance, we

should minimize $\|w\| = w^T w$ with the condition that there are no data points between HP_1 and HP_2 $w \cdot x - b \geq +1$, for positive example $y_i = +1$ and $w \cdot x - b \leq -1$, for negative example $y_i = -1$

These two condition can be combined into: $y_i(w \cdot x - b) \geq 1$

Now the problem can be formulated as

$$\min_{w, b} \frac{1}{2} w^T w \quad \text{subject to} \quad y_i(w \cdot x - b) \geq 1 \quad (9)$$

This is a convex, quadratic programming problem (in w, b) in a convex set.

Introducing Lagrange multipliers $\alpha_1, \alpha_2, \dots, \alpha_n \geq 0$, we have the following Lagrangian:

$$L(w, b, \alpha) \equiv \frac{1}{2} w^T w - \sum_{i=1}^N \alpha_i y_i (w \cdot x_i - b) + \sum_{i=1}^N \alpha_i \quad (10)$$

2.4.2 Non Linear SVM

When the two classes are non-linearly distributed then SVM can transform the data points to another high dimensional space such that the data points will be linearly separable. Let the transformation be $\Phi(\cdot)$. In the high dimensional space, we solve

$$L_D \equiv \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \Phi(x_i) \cdot \Phi(x_j) \quad (11)$$

Suppose, in addition, $\Phi(x_i) \cdot \Phi(x_j) = k(x_i, x_j)$. That is, the dot product in that high dimensional space is equivalent to a kernel function of the input space. So, we need not be explicit about the transformation $\Phi(\cdot)$ as long as we know that the kernel function $k(x_i, x_j)$ is equivalent to the dot product of some other high dimensional space.

The Mercers's condition can be used to determine if a function can be used as a kernel function:

There exists a mapping Φ and an expansion

$$K(x, y) = \sum_i \Phi(x)_i \Phi(y)_i \quad (12)$$

if and only if, for any $g(x)$ such that $\int g(x)^2 dx$ is finite, then

$$\int K(x, y) g(x) g(y) dx dy \geq 0. \quad (13)$$

There are many kernel functions that can be used this way, for example:

2.4.2.1 Polynomial

A polynomial mapping is a popular method for non-linear modeling,

$$\begin{aligned} K(x_i, x_j) &= \langle x_j, x_j \rangle^d \\ K(x_i, x_j) &= (\langle x_j, x_j \rangle + 1)^d \end{aligned} \quad (14)$$

The second kernel is usually preferable as it avoids problems with the Hessian becoming zero.

2.4.2.2 Gaussian Radial Basis Function

Radial basis functions have received significant attention, most commonly with a Gaussian of the form,

$$K(x_i, x_j) = \exp \left(- \frac{\|x_i - x_j\|^2}{2\sigma^2} \right) \quad (15)$$

Classical techniques utilizing radial basis functions employ some method of determining a subset of centers. Typically, a method of clustering is first employed to select a subset of centers. An attractive feature of the SVM is that this selection is implicit, with each support vectors contributing one local Gaussian function, centered at that data point.

2.4.2.3 Exponential Radial Basis Function

A radial basis function of the form,

$$K(x_i, x_j) = \exp \left(- \frac{\|x_i - x_j\|^2}{2\sigma^2} \right)$$

(16)

produces a piecewise linear solution which can be attractive when discontinuities are acceptable. SVM also consider some other kernels e. g., multi-layer perceptron, Fourier series, splines etc.

3 Experiments

Our motivation is to analyze how much the different dataset classification accuracy depends on the hyper parameters of SVMs. In this paper, we focus on SVMs with linear, polynomial and radial basis kernels. The most important criterion for evaluating the performance of these classifiers is the classification accuracy and the computational time. The experiments were done on two sets of data. The noise datasets {smo, thy, wav} were collected from Loh [21] and remaining from UCI repository [22].

Table 1. Datasets and splitting amount

Datasets	Size in KB	Attributes	Train Instances	Test Instances
dna	239	60C	2000	608
german	52	24C	1000	333
pendigit-8	491	16C	7494	2290
smonoise	120	12C 3B	1855	562
thynoise	412	25C 10B	3772	1145
wavnoise	618	40C	3000	910

A split strategy was used for applying the prediction methods. The training set (70%) and test set (30%) were created randomly. The description of the datasets is presented in Table 1 (C = continuous and B = binary). We used a Pentium III, 933 MHz processor with 256 MB RAM. We considered the suitable ranges of kernel parameter values after a trial and error approach. We normalized the training and test data sets, as data normalization is required for some kernels due to restricted domain, and may be advantageous for unrestricted kernels. Additionally, normalization also improves the condition of the Hessain in an optimization situation. We have chosen Navie Bayes and Neural Network from Weka data mining tool. We considered 100 runs for error minimization. We used Unix to simulate the C4.5 decision tree algorithm and Matlab for SVM programming.

4 Results and Discussions

We have illustrated the classification accuracy and computational complexity of the different learning algorithms in Table 2 and 3. Accuracy is used as a measure of performance and computational time is used as a measure of classifier speed. Figures in boldface indicate the highest accuracy in Table 2. Overall, SVMs show the highest number of accuracy for most of the datasets. SVM (polynomial kernel) shows the highest accuracy for dataset german,

wavnoise and smonoise with value of $d = 4, 2$ and 5 respectively. For the pendigit-8 data set SVM (RBF kernel) with $\gamma = 3$ performed better in terms of accuracy. The values of bias for linear kernel were set as $\{-3.01, 0.423, 3.637, -0.901, \text{ and } -2.586\}$. SVM (polynomial) with degree 5 performed better for most of the experiments. Figure 3 shows the comparative individual performance of each classifier on terms of percentage of accuracy. Figure 4 shows the percentage of accuracy for each dataset for all the classifiers considered.

Taking into account of the training time, SVMs are faster then neural networks but they are more computationally expensive than Navie Bayes and C4.5. Figure 5 illustrates the total time (learning and testing) for each dataset for building and evaluating the models. For clarity, the computational time for neural network has been reduced (scaled) 100 times and SVM (polynomial and RBF) 10 times less in Figure 5.

Table 2. Performance evaluation of classification accuracy

Datasets	Naive Bayes	C4.5	NN	SVM (linear)	SVM(polynomial) degree $d =$				SVM (RBF) width $\gamma =$				
					2	3	4	5	0.6	1.2	2	3	3.5
dna	84.83	94.83	82	79.17	84.31	85.32	85.84	85.75	69.14	61.8	59.94	59.61	59.19
german	73.33	71.33	71	79.27	87.69	96.7	99.7	99.1	90.99	93.69	97.89	98.79	98.79
pendigit-8	86.27	98.76	99.62	96.11	99.57	99.6	99.66	99.63	99.77	99.87	99.91	99.94	99.87
smonoise	70.32	63.13	59.53	71.88	71.88	76.69	87.9	92.88	71.88	72.42	76.33	80.78	82.56
thynoise	94.61	99.38	91.96	93.45	95.54	98.95	98.69	98.69	92.48	94.23	96.33	97.38	97.38
wavnoise	85.89	84.89	89.56	89.5	92.66	85.33	85.5	85.83	91.66	91.83	92	87.33	83.16
Average	82.54	85.39	82.28	87.74	89.91	90.14	92.94	94.26	88.95	89.59	91.14	91.36	90.74

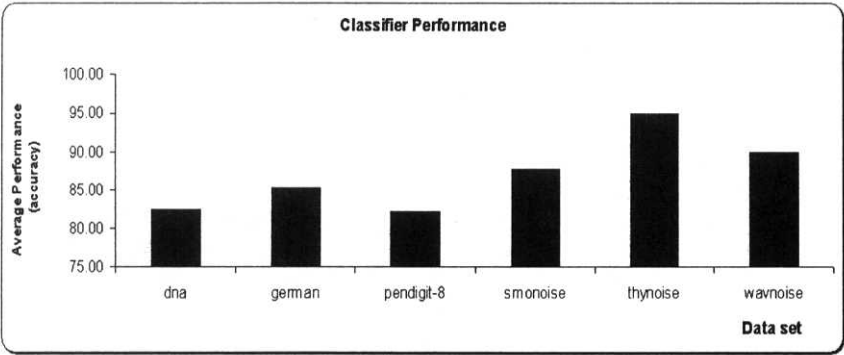


Figure 3. Classifiers average performance (accuracy %)

Table 3. Computational time for classifiers

Datasets	NBayes	C4.5	NN	SVM (linear)	SVM (poly)	SVM (rbf)
dna	0.28	1.04	10054.5	11.064	14.971	23.965
german	0.05	0.35	934	1.261	6.57	1.992
pendigit-8	0.14	0.6	1656.4	6.288	2.304	8.712
smonoise	0.07	0.74	750.9	3.365	29.651	6.019
thynoise	0.35	0.4	6758.2	4.936	20.99	59.396
wavnoise	0.29	1.86	6959.6	7.49	9.935	12.368

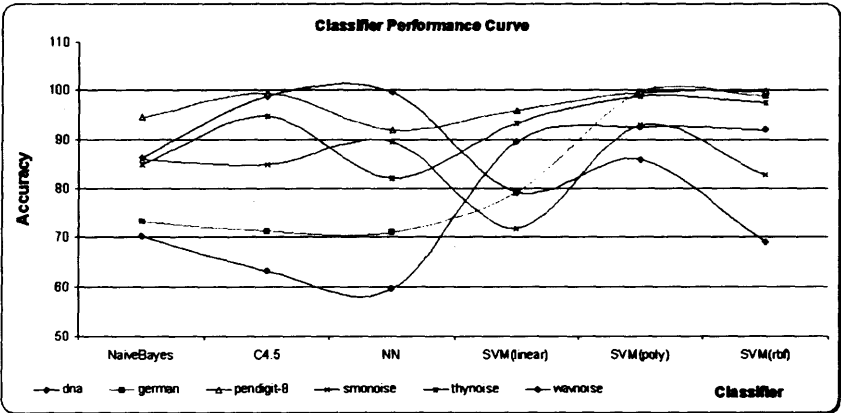


Figure 4. Classifier performance for all the data sets

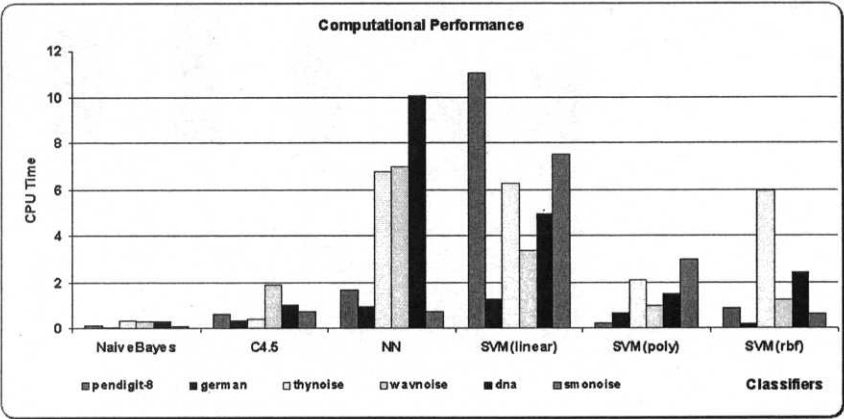


Figure 5. Computational time for different classifiers

5 Conclusions and Future Directions

In this paper, we have presented the performance of SVMs using different kernels on 6 different data sets and a comparison was made with neural networks, decision trees and Naive Bayes algorithm. From the empirical results, we can conclude that SVMs are an excellent approach to solve classification problems. SVM polynomial kernel provided the best performance than other kernels. The number of SVs selected by the SVM learning algorithm was usually small, even for very large datasets and the resulting SVM was consequently very efficient. However, using a conventional optimizer to train SVM is not the ideal solution. The selection of user defined kernel parameters is not an easy task especially for complicated problems. It is observed that the choice of these parameters has a significant effect on the generalization performance of the classifier.

Evolutionary computation has been widely adopted to optimize neural networks [23] and fuzzy inference systems [24]. Our future research will be organized to develop evolutionary computation based frameworks for automatic adaptation of the type of kernels and its parameters according to the problem environment. The optimal type of kernel and the parameter value could be formulated as a hierarchical evolutionary search as depicted in Figure 4.

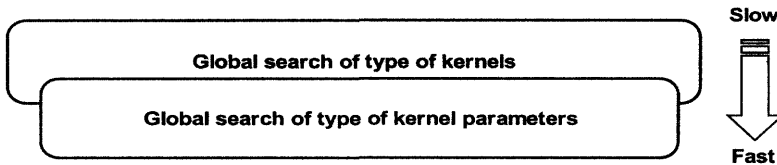


Figure 6. Interaction of various search mechanisms for the design of optimal SVMs

The lessons learned from the optimization of neural networks and fuzzy inference systems clearly reveal that an optimal SVM can only be achieved by the adaptive evolution of suitable kernel functions and its parameters that progress on different time scales. Figure 6 illustrates the general interaction mechanism with the global search of kernel functions evolving at the highest level on the slowest time scale. For each kernel function (polynomial, Gaussian and so on), global search of kernel parameters proceeds at a faster time scale in an environment decided by the problem.

References

- [1] Vapnik V., *The Nature of Statistical Learning Theory*, N.Y., Springer, 1995.
- [2] Cherkassky V. and Mulier F., *Learning from Data Concepts, Theory and Methods*, John Wiley & Sons, NK, 1998.
- [3] Gradley P. S. and Mangasarian O. L., *Massive Data Discrimination via Linear Support Vector Machines, Computational Optimization methods and Software*, 13:1-10, 2000.
- [4] Burges C. J. C., *A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery*, 2(2): pp. 121-167, 1998.
- [5] Mangasarian O. L., *Generalized Support Vector Machines*, In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pp 135-146, Cambridge, MA 2000.
- [6] Ali A. B. M. S., *Support Vector Machine and Neural Network with Others for Two Dimensional Classification Problem: An Empirical Study*, Proceedings of the International Workshop on Intelligent Knowledge Management Techniques (IKOMAT'02), IOS Press, Amsterdam, The Netherlands, 2002. (forth coming).
- [7] Amari S. and Wu S., *Improving Support Vector Machine Classifiers by Modifying Kernel Functions, Neural Networks*, v.12, pp.783-789, 1999.

- [8] Scholkopf B., Sung K. K., Burges C. J. C., Girosi F., Niyogi P., Poggio T., and Vapnik V., Comparing Support Vector Machines with Gaussian Kernels to Radial Basis Function Classifiers, *IEEE Transactions on Signal Processing*, Vol. 45, No. 11, November 1997.
- [9] Scholkopf B., Mika S., Burges C. J. C., Knirsch P., Muller K. R., Ratsch G. and Smola A. J., Input Space Versus Feature Space in Kernel-Based Methods, *IEEE Transactions on Neural Networks*, Vol. 10, No. 5, September 1999.
- [10] Scholkopf B., Taylor J., S., Smola A. J. and Williamson R. C., Kernel-Dependent Support Vector Error Bounds, *IEEE conference on Artificial Neural Networks*, 7-10 September 1999.
- [11] Joachims T., Text Categorization with Support Vector Machines: Learning with Many Relevant Features, *Proceedings of 10th European Conference on Machine Learning ECML-98*, pp. 137-142, 1998.
- [12] Scott M. J. J., Niranjani M. and Prager R. W., Parcel: Feature Subset Selection in Variable Cost Domains, *CUED/F-INFENG/TR.323*, Cambridge University, UK., 1994.
- [13] Gama J. M. P., *Combining Classification Algorithms* Vol.13 No. 2, IOS Press, 2000.
- [14] Quinlan J. R., C4.5: Programs for Machine Learning, *Morgan Kaufmann*, San Mateo, CA., 1993.
- [15] Kennedy R. L., Lee Y., Roy B. V., Reed C. D. and Lippman R. P., *Solving Data Mining Problems Through Pattern Recognition*, Prentice Hall, USA, 1998.
- [16] Aronszajn N., Theory of Reproducing Kernels, *Trans. Amer. Math. Soc.* 686, pp. 337-404, 1950.
- [17] Girosi F., An Equivalence Between Sparse Approximation and Support Vector Machine, *A.I. Memo 1606*, MIT Artificial Intelligence Laboratory, 1997.
- [18] Heckman N., The theory and Application of Penalized Least Squares Methods or Reproducing Kernel Hilbert Spaces Made Easy, Technical Report, Department of Statistics, The University of British Columbia, 1997.
- [19] Wahba G., Spline Models for Observational Data, *Philadelphia: series in Applied Mathematics*, Vol. 59, SIAM, 1990.
- [20] Platt J. C., Sequential Minimal Optimization for SVM, <<http://www.datalab.uci.edu/people/xge/svm/smo.pdf>> 2001.
- [21] Loh. W., Knowledge Discovery Central, Data Sets, <<http://www.KDCentral.com/>>, 2002.
- [22] Blake C. and Merz C. J., *UCI Repository of Machine Learning Databases*. <http://www.ics.uci.edu/~mlearn/MLRepository.html>, Irvine, CA: University of California, 2002.
- [23] Abraham A., Optimization of Evolutionary Neural Networks Using Hybrid Learning Algorithms, 2002 IEEE International Joint Conference on Neural Networks (IJCNN'02), Volume 3, pp. 2797-2802, 2002.
- [24] Abraham A. and Nath B., Evolutionary Design of Fuzzy Control Systems - An Hybrid Approach, In *Proceedings of The Sixth International Conference on Control, Automation, Robotics and Vision*, (ICARCV 2000), (CD ROM Proceeding), Wang J.L. (Editor), ISBN 981 0434456, Singapore, 2000.

Adaptive Support Vector Classifications

Zhenqiu Liu¹ and Ying Xu²

The University of Tennessee and Oak Ridge National Laboratory, USA*

Abstract. Support vector machines (SVM) was originally designed for regression and binary classification. It promises to give good generalization and has been applied to various tasks. The basic idea behind SVM is to do the classification through solving a nonlinear (quadratic) programming. In this paper, we concentrate on adaptive support vector classification problems. Since there are many parameters in the kernel functions of SVM, tuning the smooth parameters can certainly improve the performance of classification. The general literature of SVM has not discussed in detail the subject of tuning the various user defined parameters. In this paper, we explore the trade-off between maximum margin and classification errors and estimate the best kernel parameters. Toy and real life data are used in the experiments.

1 Introduction

A general two class pattern classification problem is posed as follows:

- Give l i.i.d. sample: $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_l, y_l)$
 where \mathbf{x}_i for $i = 1, \dots, l$ is a column vector of length d and $y_i = \{+1, -1\}$ is the class label for data point \mathbf{x}_i .
- Find a classifier with the decision function, $f(\mathbf{x})$ such that $y = f(\mathbf{x})$, where y is the class label for data point \mathbf{x} .

The performance of the classifier is measured in terms of classification error which defined in equation (1)

$$E(y, f(\mathbf{x})) = \begin{cases} 0 & \text{if } y = f(\mathbf{x}) \\ 1 & \text{otherwise.} \end{cases} \quad (1)$$

Evgeniou and his coworkers (2000) gave a nice explanation for the SVM from the regularization theory point of view and provided the following general function form

$$H[f] = \frac{1}{l} \sum_{i=1}^l E(y_i, f(\mathbf{x}_i)) + C \|f\|_K^2, \quad (2)$$

where $E(\cdot, \cdot)$ is an *error function*, also called *energy function*, or *loss function*, $\|f\|_K^2$ is a norm in a Reproducing Kernel Hilbert space H defined by a positive definite function K , l is the number of data points or examples, and C is the regularization parameter. They showed that both regression and classification in SVM correspond to the minimization of H in equation (2) for different choices of E :

*Contact author: Zhenqiu Liu, CHRG, The University of Tennessee, Knoxville, TN 37996, USA. email: zliu@utk.edu.

- Classical (L_2) Regularization Networks (RN)

$$E(y_i, f(\mathbf{x}_i)) = (y_i - f(\mathbf{x}_i))^2 \quad (3)$$

- Support Vector Machines Regression (SVMR)

$$E(y_i, f(\mathbf{x}_i)) = |y_i - f(\mathbf{x}_i)|_\epsilon \quad (4)$$

- Support vector Classification (SVMC)

$$E(y_i, f(\mathbf{x}_i)) = |1 - y_i f(\mathbf{x}_i)|_+ \quad (5)$$

where $|\cdot|_\epsilon$ is Vapnik's epsilon-insensitive norm (Vapnik 1995), $|x|_+ = x$ if x is positive and zero otherwise, and y_i is a real number in RN and SVMR, whereas it takes value $-1, 1$ in SVMC. Error function (5) is also called the *soft margin* error (loss) function. In this paper we only concentrate on the classification problem. SVMC can be formulated as the problem of minimizing :

$$H[f] = \frac{1}{l} \sum_{i=1}^l |1 - y_i f(\mathbf{x}_i)|_+ + \frac{1}{2C} \|f\|_K^2, \quad (6)$$

which is again of the form (2). Using the fact that $y_i \in \{-1, 1\}$ it is easy to see that our formulation in equation (6) is equivalent to the following quadratic programming problem (evgeniou et al, 2000):

$$\min \Phi(f, \xi) = \frac{C}{l} \sum_{i=1}^l \xi_i + \frac{1}{2} \|f\|_K^2$$

subject to:

$$\begin{aligned} y_i f(\mathbf{x}_i) &\geq 1 - \xi_i, & i = 1, \dots, l \\ \xi_i &\geq 0 & i = 1, \dots, l. \end{aligned} \quad (7)$$

The solution to this problem is again of the form:

$$f(\mathbf{x}) = \sum_{i=1}^l \alpha_i K(\mathbf{x}_i, \mathbf{x}) - b, \quad (8)$$

where it turns out that $0 \leq \alpha_i \leq \frac{C}{l}$. The input data points \mathbf{x}_i for which α_i are different from zero are called *support vectors*.

If we take a 2-norm $\|f\|_2 = \sqrt{\sum_{i=1}^n f_i^2}$, and let C substitute $\frac{C}{l}$ (since it is only a constant), we can formulate (7) as follows:

$$\min C \sum_{i=1}^l \xi_i + \frac{1}{2} \|f\|_2^2$$

subject to:

$$\begin{aligned} y_i f(\mathbf{x}_i) &\geq 1 - \xi_i, & i = 1, \dots, l \\ \xi_i &\geq 0 & i = 1, \dots, l. \end{aligned} \quad (9)$$

The solution is in the form of $f(\mathbf{x}) = \sum_{i=1}^l y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) - b$. The dual of (9) is as follows:

$$\min \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^l \alpha_i$$

subject to:

$$\begin{aligned} \sum_{i=1}^l y_i \alpha_i &= 0 \\ 0 \leq \alpha_i &\leq C, \quad i = 1, \dots, l. \end{aligned} \quad (10)$$

The resulting *support vector classifier* is

$$f(\mathbf{x}) = \text{sign}\left\{\sum_{i=1}^l \alpha_i^* y_i K(\mathbf{x}, \mathbf{x}_i) - b\right\}, \quad (11)$$

where it only depends on support vectors ($\alpha_i^* > 0$).

There are many algorithms based on either solving equation (9) or equation (10). In practice the dual problem is usually solved for computational efficiency. This is a quadratic problem with l variables, 1 constraint and l bounds. We may solve the equations using math programming solver directly or some new simple algorithms without using the solver. The most popular one without using the solver is called sequential minimal optimisation (SMO) algorithm. We will use it in our computations.

In support vector classifications, all the parameters of kernel functions and soft margin parameter C are predetermined. They are given without considering the specific data set. In practice, however, these tunable parameters need to be determined in the computing process in order to achieve the best generation results. There are only two major parameters which are defined by the user. There is the trade off between the margin width and the classification error (soft margin parameter C), and the kernel function parameters. The kernel parameters define the structure of the high dimensional feature space where the maximal margin hyperplane is found. Too rich a feature space would cause the system overfit the data, and conversely the system can be unable to separate the data if the kernels are too poor. Two popular kernel functions are

1. Polynomial kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (x_i \cdot x_j)^p$$

2. Radial Basis Function (RBF) kernels:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{(2\sigma^2)}\right]$$

Therefore, we need to tuning parameter σ or p alternatively for RBF or polynomial kernel.

Most papers published are dealt with soft margin parameters C (R, Herbrish and J, Weston, 1999, 2000; K, Tsuda, 1999). In this paper we will discuss the effect of different parameters and the performances of different measures (criteria). This paper is organized as follows. In section 2, we introduce different performance measures (criteria) to be used in the computational experiments. The computational results are given in section 3. Conclusions and remarks are discussed in section 4.

2 Performance Measures

The parameters are usually tuned by minimizing the estimated generalization error using some methods of performance measures such as k-fold cross validation or leave one out (LOO).

2.1 VC Bound

SVMs are based on the idea of structural risk minimization introduced by statistical learning theory (Vapnik, 2000). suppose there is a learning machine with adjust parameters α . For the two-class classification problem, the learning machine will tune its parameters α to learn the mapping $\mathbf{x} \rightarrow f(\mathbf{x}, \alpha)$. The performance of this machine (model) can be measured by the expectation of the test error, as shown in equation (12)

$$R(\alpha) = \int \frac{1}{2} E(y, f(\mathbf{x}, \alpha)) dP(\mathbf{x}, y). \quad (12)$$

This is called expected risk or actually risk. It requires at least an estimate of probability distribution $P(\mathbf{x}, y)$, which is not available for most classification tasks. Hence, one must settle for the *empirical risk* measure which is defined in equation (13). This is just a measure of the mean error over the available training data.

$$R_{emp}(\alpha) = \frac{1}{2l} \sum_{i=1}^l E(y, f(\mathbf{x}, \alpha)) = \frac{1}{2l} \sum_{i=1}^l |y - f(\mathbf{x}, \alpha)|_+. \quad (13)$$

Most training algorithm for learning machines implement Empirical Risk Minimization, i.e. minimize the empirical error using Maximum Likelihood Estimation of the parameters α . The traditional training algorithms do not consider the capacity of the learning machine and can result overfitting.

In contrast with ERM, the goal of *Structural Risk Minimization* (SRM) (vapnik, 1995, 2000) is to find the learning machine that yields a good trade-off between low empirical risk and small capacity. For particular choice of α , with probability $1 - \eta$ ($0 \leq \eta \leq 1$), the following bound holds:

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(\log(2l/h) + 1) - \log(\eta/4)}{l}}. \quad (14)$$

where h is the VC-dimension of a set of functions $f(\mathbf{x}, \alpha)$ and it describes the capacity of the set of functions. The right hand side of (14) is referred as *risk bound*. The second term of the risk bound is usually referred as the *VC confidence*

For a given learning task, the *Structural Risk Minimization* principle chooses the parameters α so that the risk bound is minimal. the main difficulty in applying the risk bound is that it is difficult to determine the VC-dimension of the set of functions. Therefore, we have to use other tools in practice.

2.2 k-fold Cross Validation and LOO

Cross validation is a popular technique for estimating generalization error and there are several versions. In k-fold cross validation, the training data is randomly split into k mutually

exclusive subsets of approximately equal size. The SVM decision rule is obtained using $k - 1$ of the subsets and then tested on the subset left out. This procedure is repeated k times and in this fashion each subset is used for test once. Averaging the test error over the k trials gives an estimate of the expected generalization error.

LOO can be viewed as an extreme form of k -fold cross validation in which k is equal to the number of examples. In LOO, one example is left out for testing each time, and so the training and testing are repeat l times. We can show :

Claim 1:

LOO estimate of the generalization error of a learning method is unbiased in that taking expected value of this estimate with respect to the probability of selecting the training set yields the true expected error.

Proof:

Let $E_D\{\cdot\}$ denote the expectations with respect to the choice of the training set. The expectation is well defined since we assume that each training example is sampled independently from some fixed but unknown distribution $P(\mathbf{x}, y)$. Now let $f(\mathbf{x}, \hat{\alpha}_l)$ be the classifier resulting from any learning algorithm that uses all l training examples. We'd like to get an estimate of the generalization error for such a classifier. LOO gives us a generalization error for almost the same thing: for $f(\mathbf{x}, \hat{\alpha}_{l-1})$, i.e., it estimates the generalization error of a classifier trained on a basis of $l - 1$ training examples. For any reasonably large training set, the two estimates are very close. Now let $f(\mathbf{x}, \hat{\alpha}_{l-1,i})$ be the classifier we obtain by training with all the training examples except the i^{th} one. With this notation, the LOO estimate of the generalization error can be written as

$$ge_{cv} = \frac{1}{l} \sum_{i=1}^l \langle y_i \neq f(\mathbf{x}_i, \hat{\alpha}_{l-1,i}) \rangle \quad (15)$$

where $\langle \cdot \rangle = 1$ if the argument is true and zero otherwise. To show that the estimate is unbiased we have to show that when averaged over possible training data sets, we get the generalization error. Consider therefore

$$E_D\{ge_{cv}\} = \frac{1}{l} \sum_{i=1}^l E_D\{\langle y_i \neq f(\mathbf{x}_i, \hat{\alpha}_{l-1,i}) \rangle\} \quad (16)$$

$$= E_D\{\langle y_i \neq f(\mathbf{x}_i, \hat{\alpha}_{l-1,i}) \rangle\} \quad (iid) \quad (17)$$

$$= E_{D_{l-1,i}}\left\{E_{(\mathbf{x}_i, y_i)}\{\langle y_i \neq f(\mathbf{x}_i, \hat{\alpha}_{l-1,i}) \rangle \mid \mathbf{x}_i, y_i\}\right\} \quad (18)$$

$$= E_{D_{l-1,i}}\{ge_{l-1,i}\} \quad (19)$$

$$= \text{Generalization error based on } l - 1 \text{ examples} \quad (20)$$

where

$$ge_{l-1,i} = E_{(\mathbf{x}_i, y_i)}\{\langle y_i \neq f(\mathbf{x}_i, \hat{\alpha}_{l-1,i}) \rangle \mid \mathbf{x}_i, y_i\}$$

This proves the Claim 1.

Claim II: The LOO cross-validation estimate for a support vector machine is upper bounded by the number of support vectors.

Proof:

Let SV be the set of support vectors. Since the solution is expressed in terms of a select set of training examples, the support vectors and all non-support vectors are classified correctly regardless of whether they appear in the training set, we may divide the summation in the

cross validation into two parts based on whether the example is a support vector or not:

$$ge_{cv} = \frac{1}{l} \left[\sum_{i \in SV} \langle y_i \neq f(\mathbf{x}_i, \hat{\alpha}_{l-1,i}) \rangle + \sum_{i \notin SV} \langle y_i \neq f(\mathbf{x}_i, \hat{\alpha}_{l-1,i}) \rangle \right] \quad (21)$$

$$= \frac{1}{l} \left[\sum_{i \in SV} \langle y_i \neq f(\mathbf{x}_i, \hat{\alpha}_{l-1,i}) \rangle + 0 \right] \quad (22)$$

$$\leq \frac{1}{l} \sum_{i \in SV} 1 \quad (23)$$

$$= \frac{\text{Number of support vectors}}{l} \quad (24)$$

Combining **Claim I** and **Claim II**, we have the following corollary.

Corollary. The expected error of a support vector machine is *upper bounded* by the expected number of support vectors.

2.3 Approximate Span Bound

Vapnik and Chapelle (1999) introduced a new concept called span of support vectors. Based on this new concept, they developed a new technique called *span rule* to approximate the LOO estimate. The following bound of LOO error estimate was also proposed in that paper.

$$\frac{N_{LOO}}{l} \leq \frac{D_{sv} \max(D_s, 1/\sqrt{C}) \sum \alpha_i^* + p}{l}, \quad (25)$$

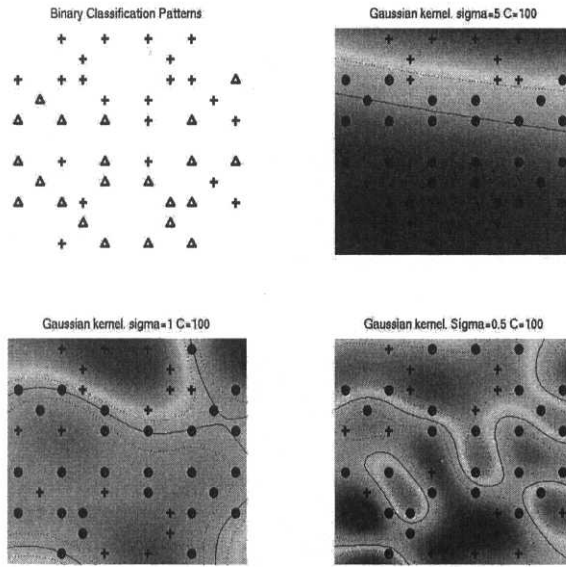
where N_{LOO} is the number of errors in LOO procedure, D_s is the diameter of the smallest sphere containing the training point in the feature space, D_{sv} is the diameter of the smallest sphere containing the support vectors that $0 < \alpha_i^* < C$, $\sum \alpha_i^*$ is the sum of support vectors that $0 < \alpha_i^* < C$, and p is the number of support vectors that $\alpha_i^* = C$. This bound is called *approximate span bound*.

3 Computational Results

The purpose of our experiments is to see how the parameters C, σ , and p affect the generalization error. and what are the best parameters for some specific data sets. All computations are based on the SOM algorithms. Both Gaussian kernel and polynomial kernel are used in the computation. In all simulations, first we fix the value of σ or p and vary the value of C , and then we fix the soft margin parameter C and vary the kernel parameter σ or p . The fixed values C and kernel parameter (σ or p) are chosen so that the the combination achieves a test error close to the smallest generalization error. In the experiments, we also calculate the margin for SVM using

$$\begin{aligned} \text{Margin} &= \frac{1}{\|\mathbf{w}\|} \\ &= \frac{1}{\sqrt{\sum_i \sum_j \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)}}. \end{aligned} \quad (26)$$

So we can explore the relation between generalization error and the classification margin.

Figure 1: Classification boundary of different σ

Our first experiment visualizes the classification boundary change according to different values of σ . A simple toy data is used for this experiment. Figure (1) shows the classification boundary of different $\sigma = \{5, 1, 0.5\}$ with fixed $C = 100$.

Figure (1) shows that the decision boundaries become more complex and irregular, when reducing σ . This greater variety in decision boundaries may add the flexibility of the model. However, if σ value is too small, overfitting will happen. We may also change the parameters p and C to reach the similar conclusions. Our next experiment gives more details in this aspect.

Our second data set is digital data set with 100 training examples and 400 test examples, where each sample has 64 dimensions and all the values are 0's or 1's. The digits in the training and test data are "3"s and "5"s. We have assigned $y = 1$ for all "3"s and $y = -1$ for all "5"s.

First we fix the value $\sigma = 0.25$, and vary the soft margin C in Equation (9). C is sometimes called error penalty. The computational results are given in table (1). Table (1) shows that the number of SVs does not change significantly with different C value. A small C does cause the average number of SVs to increase slightly. This could be due to more support vectors being needed to compensate the bound on the other support vectors. The margin increases with smaller C . This is as expected, because if errors are allowed, then the training algorithm can find a separating plane with much larger margin. The training error increases as C decreases. This is reasonable, since smaller C indicates a small search space for the QP optimizer and causes training error increase. However, the testing error decreases first and then increases when C gets smaller. The best C for a given $\sigma = 0.25$ is therefore in the range $0.6 - 0.75$ with the lowest test error 8.75%.

Second, we fix the $C = 10$ and vary the polynomial kernel parameter p . The computing

Table 1: Computing Results for Different Penalty (C)

Penalty C	Training Error	Test Error	Num SVs	Margin
100	0	9.75%	96	0.1362
10	0	9.75%	96	0.1362
1	0	9.5%	96	0.1428
0.75	1%	8.75%	96	0.1497
0.6	1%	8.75%	96	0.1591
0.5	1%	9.0%	98	0.1705
0.4	1%	12%	99	0.1922
0.3	1%	13.25%	99	0.2277

Table 2: Computing Results for Different p

Polynomial degree d	Training Error	Test Error	Num SVs	Margin
1	0	13.8%	35	0.104
2	0	10.0%	77	0.142
4	0	7.8%	98	0.128
8	0	13.8%	100	0.107

results are in table (2).

Third, we fix the $C = 10$ and vary the gaussian kernel parameter σ . The computing results are in table (3)

Table (2) shows the lowest test error is 7.8% when the polynomial degree p is 4. Table (3) shows the lowest test error is 9.8% when $\sigma = 0.25$. It is clear that when the polynomial degree of the kernel increases or when the radius of the gaussian kernel becomes smaller we can represent greater variety of decision boundaries (more complex model). Initially the added flexibility helps the generalization error as the model can better fit the data. However, eventually this leads to overfitting as we see a little bit in polynomial kernel of degree 8 and surely with gaussian kernel of radius $1/16$.

The relationship between the test error and number of SVs is ambiguous. For many of the model parameters, the number of SVs actually has an inverse relationship with the test error! Why is this? As we determined in the previous section, The number of SVs is an *upper bound on average* on the generalization error for a model. It says nothing about how close the generalization error is to the number of SVs. Hence there is no real reason to expect that the generalization error will closely follow the number of SVs.

The relationship between margin and test error of the model is a little bit clear. We can

Table 3: Computing Results for Different σ

Radius (σ)	Training Error	Test Error	Num SVs	Margin
8	0	12.8%	39	0.038
4	0	12.3%	41	0.054
2	0	11.8%	45	0.078
1	0	11.5%	55	0.107
1/2	0	11.0%	76	0.135
1/4	0	9.8%	96	0.136
1/9	0	10.0%	100	0.109
1/16	0	19.3%	100	0.101

Table 4: Test error of different criteria for optimal σ

Best Values	Banana ($C = 180$)	Image ($C = 55$)	Splice ($C = 1.5$)
Optimal σ_{opt}	2.0	3.20	50
Test Error	10.4%	1.90%	9.47%
VC Bound	40.94%	25.62%	17.7%
Span Bound	39.4%	14.5%	14.1%
5-fold CV	12.8%	1.98%	9.75%

Table 5: Test error of different criteria for optimal C

Best Values	Banana ($\sigma = 1.35$)	Image ($\sigma = 1.65$)	Splice ($\sigma = 5.5$)
Optimal C	5.2	4.3	0.4
Test Error	10.4%	1.78%	9.47%
VC Bound	39.9%	15.8%	48%
Span Bound	12.5%	5.35%	11.4%
5-fold CV	12.8%	1.98%	9.5%

see to some degree that the margin can serve as an indicator of the generalization error. In both the polynomial kernel and the gaussian kernel, the maximum margin with $p = 2$ or $\sigma = \{1/2, 1/4\}$ matches or is close to the maximum test error. However, the margin that we can achieve on the training set is affected by many factors such as C and thus larger margins need not lead to better generalization.

Our third experiments using some bench mark data sets given by G. Ratsch (1999). The first data set called *Banana* has 2 input variables, 400 training, and 4900 test examples. The second data set called *Image* has 18 variables, 1300 training, and 1010 test examples. The third data set named *splice* has 60 variables, 1000 training, and 2175 test samples. For fixed C 's, the computing results for optimal σ and different bounds are shown in table (4)

The experiment results for fixed σ 's, and Optimal C 's of three different data sets is shown in table (5).

Table (4) and (5) show that 5- fold cross-validation gives an excellent estimate of the generalization error. Non of others yields a performance as good as 5-fold cross validation. One the other hand, the approximate span bound and VC bound cannot give a useful prediction of the optimal parameters. We believe this is because the approximations introduced into these bounds are too loose.

Our final example is a genetic classification problem. The data set for threading scores have 36657 samples and only 1551 true pairs. The data set is in 20 dimensional space. The data set is unbalanced in the sense that the true pairs are only a small proportion (4.23%). Moreover, The data set is also linear inseparable as discussed in the original paper. and the objective of our training is how to recognize the true pairs from huge false pairs. One way to classify the data set is to classify all the samples to be false pairs and the training and test error can be both under 5% !! However, we know this is wrong as this classifier does not provide any information to recognize the true pairs. Therefore, the low training or test error does not indicate a good performance in this special problem. One way to deal with this problem is to use the weighted error function. i.e., to add more punishment for the error that have a true pair classified as a false pair. Another way to deal with this problem is to choose the training data set carefully using some feature selection methods. the second approach is used in this paper.

Table 6: Computing Results using Different Models

Models	Training Error	Test Error	training/real	test/real
Linear	0.025	0.043	298/800	56/751
Quadratic	0.021	0.047	376/800	0/751
SVM	0.020*	0.032*	348/800	204/751

The comparison of different model performance are given in Table (6). All experiments are using the same training sample set with 20419 patterns.

Then, why SVM has the better performance? One of the explanation could be that the soft margin SVM have the regularization term C , so it can deal with inseparable data set efficiently.

4 Conclusions

Support vector machine is a powerful tool in solving classification and regression problems. Adapting related parameters in SVMs can improve the performance of SVM and reduce the generalization (test) error. We showed that the maximal margin has a more positive relation with the test error than the number of support vectors does. We also exhibited that reducing σ , decreasing C , or increasing p would increase the complexity of the model and change the test error. Among all of the popular measures tested, LOO and k-fold cross validation have the better performance. Future works may include multi-class SVMs and feature selection for SVMs.

References

- [1] H. Bozdogan and Z. Liu, RBF neural networks for classification using new kernel functions, *Intelligent System and Applications*. Dynamic Publisher (2002) 17–22.
- [2] N. Cristianini and C. Campbell, Dynamically Adapting kernels in Support Vector Machines, *Proceedings of ICML2000*. Stanford, CA, (2000).
- [3] T. Evgeniou, M. pontil, and T. Poggio, Regularization networks and support vector machines. *Advances in Computational Mathematics* 13 (2000) (1):1–50.
- [4] Z. Liu, *Intelligent Data Mining using Kernel Functions and Information Criteria*, PhD thesis, University of Tennessee (2002).
- [5] E. Osuna, *Support Vector Machines: Training and Applications*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, (1998).
- [6] J. Platt, Sequential minimal optimization: A fast algorithm for training support vector machines, In Scholkopf et al., pages 185–208. <http://www.research.microsoft.com/jplatt/smo.html>. Springer, (1999).
- [7] V. Vapnik and O. Chapelle, Bounds on error expectation for support vector machine, in Smola ,Bartlett, Scholkopf and Schuurmans.Ed. *Advances in Large Margin Classifiers*, MIT Press,(1999).
- [8] J. Weston and R. Herbrich. Adaptive margin support vector machines. In A. J. Smola, P. L. Bartlett, B. Scholkopf, and Schuurmans, editors. *Advances in Large margin Classifiers*, pages Cambridge, MA, MIT Press, (2000) 281–296.

Mercer's Kernel Based Learning for Fault Detection

Bernardete Ribeiro and Paulo Carvalho

Centre of Informatics and Systems,

Department of Informatics Engineering,

University of Coimbra

Pólo II, Pinhal de Marrocos, 3030 Coimbra, Portugal

bribeiro@dei.uc.pt, carvalho@dei.uc.pt

Abstract. This paper proposes a Mercer's kernel based learning for classification problems using Minkovsky's norm. A comprehensively comparative study of the main characteristics of the support vector algorithm using various values of α parameter in norm's definition is presented. Special emphasis is laid on kernel machine accuracy evaluation and model complexity using Gaussian kernels. Experimental results are given concerning a real application dealing with classification of part defects in an injection molding machine for plastics industry. Also, future research directions are outlined. Keywords: Kernel learning, Support Vector Machines, Fault Detection.

1 Introduction

In the realm of machine learning one long-standing goal addresses the crucial question of how well the sought estimated learning function generalizes. The availability of reliable learning systems is of major importance since in most cases no mathematical model is known and thus classical programming is not appropriate. In past years, research efforts have focused on building a universal class of models (such as neural networks) generally applicable to a broad scope of areas. Recently, support vector machines (SVMs) [1] span a wide range of classification, regression and density estimation problems and many applications have been reported [2][3][4]. This approach has its roots in statistical learning theory [5] and provides a way to build "optimum classifiers" according to some optimality criterion that is referred to as the maximal margin criterion. While traditional inferential statistics operate on a-priori hypotheses tested on data sets collected ideally after the hypothesizing and specifically for testing the hypothesis, SVMs find an optimal model based upon existing data sets. Besides, while traditional statistics assumed that the correct model was known and the task was to estimate parameters, statistical learning theory focused on comparing the relative performance of competing models with the assumption that the correct model is unknown. This theory also uses finite-sample statistics rather than asymptotic statistics. An interesting development in statistical learning theory is the replacement of the degrees of freedom concept with the Vapnik-Chervonenkis (VC) dimension, which is a measure of the complexity of the model.

Support vector theory, equipped with a sound mathematical background (see for example [6]), deals with how to minimize complexity in the course of learning and how high generalization can be attained. This trade-off between complexity and accuracy has lead to various

principles to find the optimal compromise so far. Although Vapnik's work can be traced back to the main elements of empirical processes theory [5], it was not until recently [7] that the generalization error has been shown to be bounded by the sum of the training error and a term depending on the Vapnik-Chervonenkis (VC) dimension of the learning machine leading to the formulation of the structural risk minimization (SRM) principle. Through minimization of this upper bound, high generalization can be achieved. These bounds will typically depend on certain quantities such as the margin of the classifier giving rise to algorithms which optimize this measure with the advantage that SVM's generalization error is not dependent on the dimensionality of the input data as in other previous approaches.

A noteworthy property of SVMs is that loss functions lead to sparse solutions. This means that, unlike regularization networks [8], only a small fraction of the coefficients in the decision function are nonzero. Another key property is the geometric interpretation of the Reproducing Kernel Hilbert Space (RKHS) norm of their solution as the inverse of the margin [1] which might explain their successful application in practical problems. Although SVMs were initially proposed for classification tasks further work extended the technique to regression problems. In [9] support vector regression is used for time series prediction and compared to radial basis functions networks. Model selection complexity is evaluated in [10] using nonasymptotic bounds on the prediction risk based on the VC theory proposed by [11]. Also very recently, support vector clustering, which deals with a boundary based rule (instead of a margin based rule) has been introduced. Results in benchmark problems and in a practical case study have been reported in [12][13].

For a given problem, the class of functions chosen by SVM is primarily determined by the kernel function k . Mercer's theorem provides a characterization of when a function is a kernel [14]. Besides, Mercer's features provide a representation of the points in input space through their image in the feature space with the inner product defined by the infinite number of the kernel eigenvalues. Although the images of the points are not computed explicitly, their inner products can be determined by the kernel function which is usually known as the kernel trick. The sub-manifold of their image in the feature space is defined by the eigenvectors of the kernel operator.

All above mentioned support vector techniques provide an useful insight of the relevant role the choice of the kernel can play. We investigate here a generalization of the Euclidean distance in a Mercer's kernel, namely, the Minkovsky's distance in Gaussian kernels for a practical case study encompassing a learned classification task problem. We compare the SVC performance for several kernel machines determined by the value of parameter α in the definition of Minkovsky's norm by cross validation.

The rest of the paper is organized as follows. In section 2 the support vector classification is briefly summarized and the optimization problem defined. In section 3 the Mercer's kernel based learning is described with emphasis put on Gaussian kernels defined with a generalized form of the Euclidean norm. Section 4 is focused on presenting and discussing experimental results in a classification practical case study concerning categorization of plastic parts's defects in an injection molding industrial process. The last section gives some concluding remarks and outlines some future directions of research.

2 Support Vector Classification

Support vector classification (SVC) has firstly been proposed for binary patterns in pattern recognition problems [15] where $y_i \in \mathcal{Y} \equiv \{\pm 1\}$ and lately extended for regression. Vapnik extended his original geometric derivation of linear SVM classifier introducing the kernel k in terms of a map into a higher dimensional feature space called a feature space ($\phi : \mathbf{x} \rightarrow \phi(\mathbf{x})$). The interpretation of the kernel simply as an inner product in feature space allowed then the forward pass from the linear separable case to the non linear separable one which is indeed more realistic in the majority of practical cases of datasets. However, in the more general framework of regularization the technique, as pointed out in [16], and according to the Representer Theorem [17] leads to solutions of the form (1):

$$f(\mathbf{x}) = \sum_{i=1}^l \alpha_i k(\mathbf{x}, \mathbf{x}_i) + b \quad (1)$$

where \mathbf{x}_i , $i = 1, \dots, l$ are the input training examples, k a symmetric positive definite function, the kernel, $\alpha_i \in \mathbb{R}$ a set of parameters to be determined from examples and $b \in \mathbb{R}$ an unregularized bias term. According to the Tikhonov regularization theory, which deals with the trade off between the training error and the complexity of the hypothesis space, the solution function f is found by minimizing functionals of the type (2):

$$\frac{1}{l} \sum_{i=1}^l V(y_i, f(\mathbf{x}_i)) + \lambda \|f\|_k^2 \quad (2)$$

where V is a loss function that measures the discrepancy of the data set and the estimated function output $f(\mathbf{x}_i)$ with respect to the desired output y_i , $\|f\|_k^2$ is the norm of the function in the Reproducing Kernel Hilbert Space (RKHS) \mathcal{F} defined by the kernel k , and λ a positive parameter. The choice of the loss function determines different learning techniques, each leading to a different learning paradigm for computing the coefficients α_i . The regularization parameter λ penalizes functions with high capacity in the sense that the higher λ the smaller the RKHS norm of the solution [18].

Several loss functions can be defined conveying to a different type of learning machine. The SVM classification arises by considering the hinge loss $V(y_i, f(\mathbf{x}_i)) = |y_i - f(\mathbf{x}_i)|_\varepsilon$ with $r = \max\{0, |r| - \varepsilon\}$, $y_i \in \mathbb{R}$. Once more, the learning machine can be formulated minimizing the functional (2) with $V(r)$ defined as the specific loss function. The equivalent quadratic programming problem originally proposed in [19] is:

$$\min_{f \in \mathcal{F}_\xi} \Phi(f, \xi) = C \sum_{i=1}^l \xi_i + \frac{1}{2} \|f\|_k^2 \quad (3)$$

subject to constraints

$$\begin{aligned} y_i f(\mathbf{x}_i) &\geq 1 - \xi_i & i = 1, \dots, l \\ \xi_i &\geq 0 & i = 1, \dots, l \end{aligned} \quad (4)$$

Introducing Lagrange multipliers:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \quad (5)$$

with respect to α_i , under the constraints where $0 \leq \alpha_i \leq C$, $i = 1, \dots, l$ and $\sum_{i=1}^l \alpha_i y_i = 0$. The solution has again the form (1). The input data points \mathbf{x}_i for which α_i is non zero are the so-called support vectors. The bias term b follows from the KKT conditions. The empirical error measured by $\sum_{i=1}^l \xi_i$ is minimized while controlling capacity measured in terms of the norm of f in RKHS. It can be shown that maximizing (5) corresponds to minimizing an upper bound of the VC dimension of separating hyperplanes, or equivalently, to maximizing the separation margin between the two classes. Considering the expansion of w above, the hyperplane decision function can be written as:

$$f(\mathbf{x}) = \text{sgn} \left(\sum_{i=1}^l \alpha_i y_i (\mathbf{x} \cdot \mathbf{x}_i) + b \right) \quad (6)$$

where b is computed by $\alpha_i (y_i (w \cdot \mathbf{x}_i + b) - 1) = 0$, $i = 1, \dots, l$, such that α_i is not zero. For a test vector $\mathbf{x} \in \mathbb{R}^N$ a class label $o(\mathbf{x})$ is then assigned by following the rule:

$$o(\mathbf{x}) = \begin{cases} 1 & f(\mathbf{x}) > 0 \\ -1 & \text{otherwise} \end{cases} \quad (7)$$

3 Mercer's Kernel Based Learning

The above formulation links up SVMs to kernel based learning algorithms where the objective of learning is to maximize the margin around the decision surface. The learning task depends on the complexity of the target function and how well the representation chosen matches a specific learning problem [20]. One common strategy in machine learning changes the representation of data in a preprocessing phase through a given mapping. Let ϕ be this feature map such that the data $\mathbf{x} \in \mathcal{X}$ (for simplicity \mathcal{X} is a compact subset $\mathcal{X} \subseteq \mathbb{R}^N$) is mapped into a higher dimensional feature space \mathcal{F} . In such a feature space, a class of kernels $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ can be used to cast the inner products computation in \mathcal{F} :

$$k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{F}} \quad (8)$$

provided Mercer's conditions [14] [21] hold which are necessary and sufficient for the existence of the mapping ϕ . In (8) $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ denotes the inner product in \mathcal{F} . The kernel function allows computations to be performed in the input space rather than in the higher dimensional feature space \mathcal{F} (the features being $\{\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \dots, \phi_{N_{\mathcal{F}}}(\mathbf{x})\}$ with $N_{\mathcal{F}} \leq \infty$ the dimensionality of the RKHS). Classification can thus be carried out in the feature space without knowing the explicit form of ϕ . A Reproducing Kernel Hilbert Space (RKHS) is a Hilbert Space of functions f of the form $f(\mathbf{x}) = \sum_{q=1}^{N_{\mathcal{F}}} a_q \phi_q(\mathbf{x})$, where $\{\phi_q(\mathbf{x})\}_{q=1}^{N_{\mathcal{F}}}$ is a set of given, linearly independent basis functions, being orthonormal eigenfunctions of the integral equation (9):

$$\int k(\mathbf{x}, \mathbf{x}') \phi(\mathbf{x}) d\mathbf{x} = \mu \phi(\mathbf{x}') \quad (9)$$

A RKHS has norm $\|f\|^2 = \sum_{q=1}^{N_{\mathcal{F}}} \frac{a_q^2}{\mu_q}$ where $\{\mu_q\}_{q=1}^{N_{\mathcal{F}}}$ is a positive sequence of real values (decreasing to 0 if $N_{\mathcal{F}} = \infty$). The eigenvalues μ_q and the basis functions $\{\phi_q(\mathbf{x})\}_{q=1}^{N_{\mathcal{F}}}$ define the symmetric positive kernel function (10):

$$k(\mathbf{x}, \mathbf{x}') = \sum_{q=1}^{N_{\mathcal{F}}} \mu_q \phi_q(\mathbf{x}) \phi_q(\mathbf{x}') \quad (10)$$

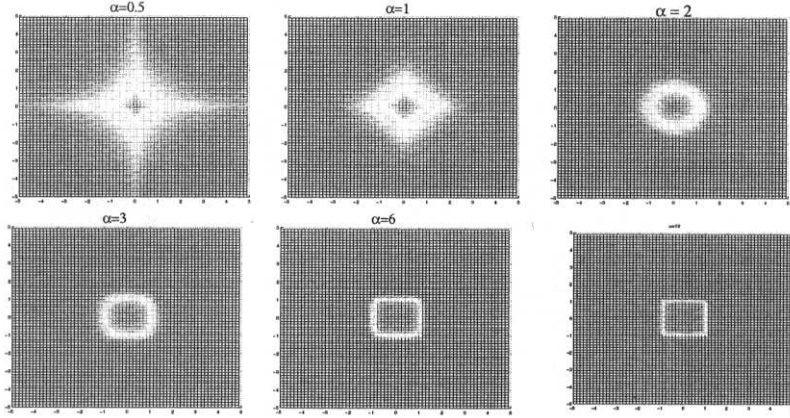


Figure 1: Gaussian Function with Minkovsky norm ($\alpha = 0.5, 1, 2, 3, 6, 10$.)

Using different kernels correspond to different functions to be chosen. There are currently no techniques available to “learn” the form of the kernels. Some common investigated kernels include polynomials of degree d ($k(\mathbf{x}, \mathbf{x}') = (1 + \langle \mathbf{x}, \mathbf{x}' \rangle)^d$), Gaussian Radial Basis Functions ($k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2)$) and Multi Layer Perceptron ($k(\mathbf{x}, \mathbf{x}') = \tanh(\kappa \langle \mathbf{x}, \mathbf{x}' \rangle - \delta)$) for suitable values of gain κ and threshold δ . In the case of a RBF kernel the number of centers (number of support vectors), the centers themselves (the support vectors), the weights (α_i) and the threshold b are calculated automatically by the SVM training. Due to the encouraging results with RBF kernels, more generalized forms based on some distance measure are suitable to be investigated. The Euclidean distance has a natural generalization in form of the Minkovsky’s distance function [22]:

$$D_M(\mathbf{x}; \mathbf{x}'; \alpha) = \left(\sum_{i=1}^N |x_i - x'_i|^\alpha \right)^{1/\alpha} \quad (11)$$

Euclidean and Manhattan distances are special cases of Minkovsky’s metric function with $\alpha = 2$ and $\alpha = 1$ respectively. Minkovsky’s distance with the scaling factors is a further generalization:

$$D_{M_b}(\mathbf{x}; \mathbf{x}'; \alpha) = \sum_{i=1}^N d(x_i; x'_i)^\alpha / b_i \quad (12)$$

The $d(\cdot)$ function is used to estimate similarity at the feature level and in the simplest case is equal to $|x_i - x'_i|$. For $\alpha = 2$ the vectors $\|\mathbf{x}\| = 1$ are on the unit sphere, for larger values of α the sphere is changed into a soft cuboid, for $\alpha = 1$ it becomes a pyramid and for $\alpha < 1$ it has a hypocycloids shape. In Figure 1 Gaussian functions with different Minkovsky distances are shown.

4 Experimental Results

4.1 Process Description

Injection molding of thermoplastics (see Figure 2) is a process where the material is heated up to reach a state of fluidity, being then injected under pressure into a mold cavity where it cools. Cooled in the mold it then reaches a solid state conforming the details of the mold producing a component part which reproduces it. Quality monitoring and optimization of the injection molding process require models which describe the complex relationships between process parameters and product properties. From historical production data it is useful to describe the process behavior when operating normally as well as to incorporate in the model abnormal behavior due to unexpected processing disturbances and/or uncorrected machine setup parameters.

Figure 2 illustrates the molding machine, a sketch of the molding functioning process and two of the observed parts defects (edges and unfilled) for the automotive industry.

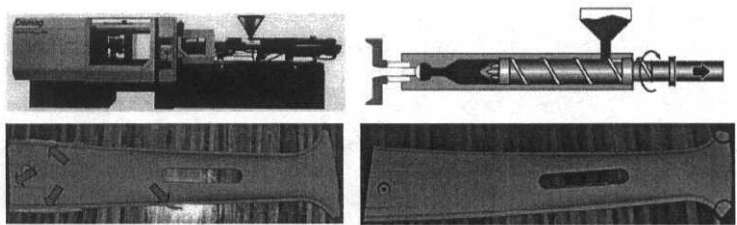


Figure 2: Molding Machine, Plasticizing Process and Plastic parts defects.

Table 1: Plastic parts identification faults.

Fault 1	Fault 2	Fault 3	Fault 4	Fault 5	Fault 6
Streaks	Stains	Burnt marks	Edges	Unfilled parts	Warped parts

4.2 Setup Experimental Conditions

The industrial data sets used for training and testing SVMs are obtained from historical production of an injection molding machine. Each pattern has six continuous inputs (cycle time, dosage time, injection time, cushion, peak melt temperature, ram velocity). The problem is to classify molded part defects into one of six classes (streaks, stains, burnt marks, edges, unfilled and warped parts).

Table 4.1 establishes the corresponding categories of observed plastic part defects. The training (761 patterns) and test (140 patterns) sets were obtained during manufacturing with plastic parts for the automotive industry from molded on a *DN502* mold and using the grade *Hostacom DM2T06*.

4.3 SVM Classification Results

The series of experiments were designed to assess the performance of the industrial kernel machine using support vector classification for finding the decision regions on our task classification problem. To solve the quadratic problem we have used the decomposition method reported in [23] and the LIBSVM software [24].

As we have a multiclass learning problem we will briefly describe the technique used herein. There are several methods for the extension from the binary two-class problem to n classes. Among the different possibilities we address the following: (i) to modify the design of SVM in order to incorporate multiclass learning in the quadratic solving algorithm [25] (ii) to combine several binary classifiers. In the second case, besides the standard approach to the n -class problem by a winner takes-all like scheme or the “one against others”, there is the pairwise classification or “one against one”. The basic idea of pairwise classification [26] is to use $n * (n - 1)/2$ classifiers covering all pairs of classes instead of using only n classifiers as in the “one against others”. This was the approach used in this paper. In fact, in this algorithm, n hyperplanes are constructed, each hyperplane separating one class from the others. In this sense, there are n decision functions $f_k (1 \leq k \leq n)$ of the form (6). Thus, the n -class classification problem can be viewed as a discriminant vector $\in \mathbb{R}^n$ where the index of the largest component is chosen as class decision.

Tables 2 and 3 present the testing results on the industrial problem investigated here for our design choice of a Gaussian kernel. Each row reports, for each α value in the Minkovsky's norm description given in (11), the number of support vectors (SV), the testing accuracy (Ac (%)) and the number of iterations (iter) using the values of $C = 1$ and $C = 1000$, for the complete range of faults observed during the injection molding process. A larger value of C achieves better performance to the classification task for the tested faults corresponding to assign a higher penalty error. In fact, the parameter C , the so-called box constraint, determines the design and performance of the SVM learning machine as long as the degree of allowed soft margin is controlled.

We examine the dependence of the algorithm's performance on the choice of parameters C and γ for a fixed numerical precision of $\varepsilon = 0.0001$. For simplification we present only the results for the hyperparameter $\gamma = 100$. While there is ongoing research for the choice of the kernel width, the great majority of applications sets γ by heuristics. Experimental results indicate that a good choice is the distance between closest points belonging to different classes, but frequently the choice must be made accordingly to the response to data. In the absence of reliable criteria, the most used techniques rely on cross validation. As observed for both tried values of C it was possible for the learning machine to enforce full separability for all faults and covered range of α parameter. In Figure 3 the influence of parameter α is depicted for the learning machine accuracy and number of iterations showing less accuracy for $\alpha = 2$. However, as the number of support vectors is lower we might expect the Mercer's kernel based learning to improve on other fault data sets.

5 Conclusion

This paper presented an industrial kernel machine for faults detection in plastics parts produced by an injection molding machine. While the main goal in the industrial framework is to automatically calculate the machine control setpoints, a less important task is to classify plastic molded parts defects efficiently in order to assess multiple quality characteristics.

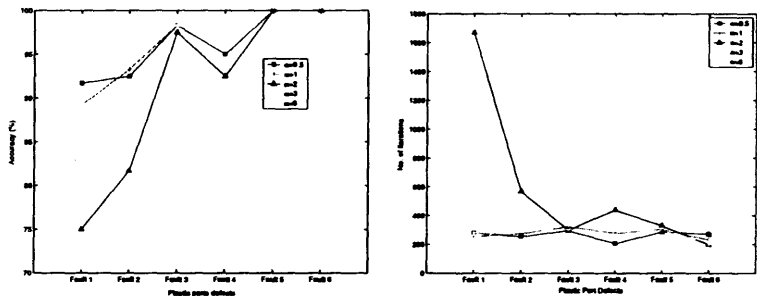


Figure 3: Performance accuracy and number of iterations of the kernel machine ($C = 1, \gamma = 100$) for values of parameter $\alpha = 0.5, 1, 2, 3, 6$.

Table 2: Kernel Evaluation ($\gamma = 100$) for varying α

Kernel		Fault 1		Fault 2		Fault 3	
$\gamma = 100$		$C = 1$	$C = 10^3$	$C = 1$	$C = 10^3$	$C = 1$	$C = 10^3$
$\alpha = 0.5$	SV	107	105	109	104	96	95
	Ac(%)	91.7	92.5	92.5	92.5	98.3	98.3
	iter	282	4232	255	4317	296	1263
$\alpha = 1$	SV	106	105	108	104	96	95
	Ac(%)	89.2	93.3	91.7	93.3	98.3	98.3
	iter	255	4216	276	4404	324	1336
$\alpha = 2$	SV	91	67	87	67	50	48
	Ac(%)	75.0	90.0	81.7	92.5	97.5	98.3
	iter	1669	47603	568	19046	300	1921
$\alpha = 3$	SV	106	105	108	104	95	94
	Ac(%)	89.2	93.3	90.8	93.3	98.3	98.3
	iter	254	3445	305	4389	308	1355
$\alpha = 6$	SV	107	105	107	104	95	94
	Ac(%)	89.2	93.3	90.8	93.3	98.3	98.3
	iter	280	3419	297	4398	305	1318

Table 3: Kernel Evaluation ($\gamma = 100$) for varying α

Kernel		Fault 4		Fault 5		Fault 6	
$\gamma = 100$		$C = 1$	$C = 10^3$	$C = 1$	$C = 10^3$	$C = 1$	$C = 10^3$
$\alpha = 0.5$	SV	95	94	95	91	91	91
	Ac(%)	95	98.3	100	100	100	100
	iter	206	1302	287	345	272	268
$\alpha = 1$	SV	97	94	93	91	91	91
	Ac(%)	95.8	98.3	100	100	100	100
	iter	278	1451	305	334	232	265
$\alpha = 2$	SV	58	54	45	44	43	43
	Ac(%)	92.5	96.7	100	100	100	100
	iter	438	4492	330	515	198	238
$\alpha = 3$	SV	96	93	92	90	90	90
	Ac(%)	95.8	98.3	100	100	100	100
	iter	274	1417	298	332	214	251
$\alpha = 6$	SV	96	93	91	90	90	90
	Ac(%)	95.8	98.3	100	100	100	100
	iter	261	1381	313	323	207	240

The proposal is based on SVMs for the underlying task classification problem. Although recent approaches have successfully applied neural networks as part quality monitoring tools, SVMs have the advantage that a smaller number of parameters for the model can be identified in a manner that does not require the extent of prior information or heuristic assumptions that some previous techniques require. Furthermore, SVMs use an algorithm which is constrained quadratic programming, a technique that has no local minima. Several Gaussian kernels used as dot products in the support vector machine each one with a different value of parameter α in the Minkovsky's norm have been tested. While the accuracy has proved to increase for larger values of α thus increasing the performance of the learning machine, the occurrence of a larger number of support vectors might decrease the generalization's machine capabilities. Further research will proceed in order to clarify the feature space properties induced by using more generalized forms of the Euclidean distance in the kernel.

References

- [1] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer Verlag, 1995.
- [2] K. Jonsson, J. Kittler, Y. Li, and J. Matas, "Support vector machines for face authentication," in *Proc. of BMVC'99* (T. Pridmore and D. Elliman, eds.), pp. 543–553, 1999.
- [3] J. T.-Y. Kwok, "The evidence framework applied to support vector machines," *IEEE Transactions on Neural Networks*, vol. 11, no. 5, pp. 1162–1173, 2000.
- [4] B. Ribeiro, "Support vector machines for fault detection and diagnosis," in *Proc. of 8th Int. Conf. on Neural Information Processing (ICONIP2001)* (L. Zhang and F. Gu, eds.), (China), pp. 873 – 879, Fudan University Press, 2001.
- [5] V. Vapnik and A. Chervonenkis, "Uniform convergence of frequencies of occurrence of events to their probabilities," *Dokl. Akad. Nauk SSSR*, vol. 181, pp. 915 – 918, 1968.
- [6] F. Cucker and S. Smale, "On the mathematical foundations of learning," *Bulletin of the American Mathematical Society*, vol. 39, no. 1, pp. 1–49, 2001.
- [7] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.
- [8] T. Poggio and F. Girosi, "Regularization algorithms for learning that are equivalent to multilayer networks," *Science*, vol. 247, pp. 978–982, 1990.
- [9] K.-R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "Using support vector machines for time series prediction," in *Proc. ICANN'97, Int. Conf. on Artificial Neural Networks* (W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, eds.), vol. 1327 of *LNCS*, (Berlin), pp. 999–1004, Springer, 1997.
- [10] V. Cherkassy, X. Shao, F. M. Mulier, and V. Vapnik, "Model complexity control for regression using vc generalisation bounds," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1075–1089, 1999.
- [11] V. N. Vapnik, *The nature of statistical learning theory*. New York: Springer-Verlag, 1995.
- [12] A. Ypma, D. Tax, and R. Duin, "Robust machine fault detection with independent component analysis and support vector description," pp. 1991–1999, 1999.
- [13] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering," *Journal of Machine Learning Research*, no. 2, pp. 125–137, 2001.
- [14] T. Mercer, "Functions of positive and negative and their connections to the theory of integral equations," *Trans. London Phil. Soc. (A)*, vol. 209, pp. 415–446, 1909.
- [15] V. Vapnik and A. Lerner, "Pattern recognition using generalized portrait method," *Automation and Remote Control*, vol. 24, 1963.

- [16] T. M. Evgeniou, M. Pontil, and T. Poggio, "Statistical learning theory: A primer," *International Journal of Computer Vision*, vol. 38, no. 1, pp. 9–13, 2000.
- [17] G. Wahba, "Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV," in *Advances in Kernel Methods — Support Vector Learning* (B. Schölkopf, C. J. C. Burges, and A. J. Smola, eds.), (Cambridge, MA), pp. 69–88, MIT Press, 1999.
- [18] T. Evgeniou and M. Pontil, "On the $V\gamma$ dimension for regression in reproducing kernel Hilbert spaces," in *Algorithmic Learning Theory: ALT-99*, Springer-Verlag, 1999.
- [19] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 273 – 297, 1995.
- [20] N. Cristianini and J. Shawe-Taylor, *Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.
- [21] R. Courant and D. Hilbert, *Methods of Mathematical Physics I*. Springer-Verlag, 1924.
- [22] W. Duch and N. Jankowski, "Survey of neural transfer functions," *Neural Computing Surveys*, vol. 2, pp. 163–212, 1999.
- [23] C.-J. Lin, "On the convergence of the decomposition method for support vector machines," tech. rep., Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2000.
- [24] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," tech. rep., Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2000.
- [25] J. Weston and C. Watkins, "Multi-class support vector machines," Tech. Rep. CSD-TR-98-04, Royal Holloway, University of London, Egham, TW20 0EX, UK, 1998.
- [26] U. H.-G. Kreel, "Pairwise classification and support vector machines," in *Advances in Kernel Methods: Support Vector Learning* (B-Schölkopf, C. Burges, and A. Smola, eds.), pp. 255–268, Cambridge, MA: MIT Press, 1999.

Performance Based Feature Identification for Intrusion Detection Using Support Vector Machines

Srinivas MUKKAMALA, Andrew H. SUNG
{srinivas, sung}@cs.nmt.edu
Department of Computer Science
New Mexico Institute of Mining and Technology
Socorro, New Mexico 87801

Abstract. Intrusion detection is a critical component of secure information systems. This paper addresses the issue of identifying important input features in building an intrusion detection system (IDS). Since elimination of the insignificant and/or useless inputs leads to a simplification of the problem, faster and more accurate detection may result. Feature ranking and selection, therefore, is an important issue in intrusion detection.

Since support vector machines (SVMs) tend to scale better and run faster than neural networks with higher accuracy for intrusion detection, we apply the technique of deleting one feature at a time to perform experiments on SVMs to rank the importance of input features for the DARPA collected intrusion data. Important features for each of the 5 classes of intrusion patterns in the DARPA data are identified.

It is shown that SVM-based IDS using a reduced number of features can deliver enhanced or comparable performance. An IDS for class-specific detection based on five SVMs is proposed.

1. Introduction

This paper mainly addresses the issue of identifying important input features for intrusion detection. Since the ability to identify the important inputs and redundant inputs of a classifier leads directly to reduced size, faster training and possibly more accurate results, it is critical to be able to identify the important features of network traffic data for intrusion detection in order for the IDS to achieve maximal performance.

Since most of the intrusions can be uncovered by examining patterns of user activities, many intrusion detection systems have been built by utilizing the recognized attack and misuse patterns to develop learning machines [1,2,3,4,5,6,7,8,9]. In our earlier work, support vector machines (SVMs) are found to be superior to neural networks in many important respects of intrusion detection [10,11,12], so we will illustrate feature-ranking using SVMs.

The data we used in our experiments originated from MIT's Lincoln Lab. It was developed for intrusion detection system evaluations by DARPA and is considered a benchmark for intrusion detection evaluations [13].

We performed experiments to rank the importance of input features for each of the five classes (normal, probe, denial of service, user to super, and remote to local) of patterns in the DARPA data. It is shown that using only the important features for classification gives good accuracies and, in certain cases, reduces the training time and testing time of the SVM classifier.

In the rest of the paper, a brief introduction to the data we used is given in section 2. In section 3 we describe the method of deleting one input feature at a time and the performance metrics considered for deciding the importance of a particular feature. In section 4 we present the experimental results of using SVMs for feature ranking. In section 5 we summarize our results and give a brief description of our proposed IDS architecture.

2. The Data

In the 1998 DARPA intrusion detection evaluation program, an environment was set up to acquire raw TCP/IP dump data for a network by simulating a typical U.S. Air Force LAN. The LAN was operated like a true environment, but being blasted with multiple attacks. For each TCP/IP connection, 41 various quantitative and qualitative features were extracted. Of this database a subset of 494021 data were used, of which 20% represent normal patterns.

Attack types fall into four main categories:

1. DOS: denial of service
2. R2L: unauthorized access from a remote machine
3. U2R: unauthorized access to local super user (root) privileges
4. Probing: surveillance and other probing

3. Ranking The Significance of Inputs

Feature selection and ranking is an important issue in intrusion detection. Of the large number of features that can be monitored for intrusion detection purpose, which are truly useful, which are less significant, and which may be useless? The question is relevant because the elimination of useless features (or audit trail reduction) enhances the accuracy of detection while speeding up the computation, thus improving the overall performance of an IDS. In cases where there are no useless features, by concentrating on the most important ones we may well improve the time performance of an IDS without affecting the accuracy of detection in statistically significant ways.

The feature ranking and selection problem for intrusion detection is similar in nature to various engineering problems that are characterized by

- Having a large number of input variables $\mathbf{x} = (x_1, x_2, \dots, x_n)$ of varying degrees of importance; i.e., some elements of \mathbf{x} are essential, some are less important, some of them may not be mutually independent, and some may be useless or noise
- Lacking an analytical model or mathematical formula that precisely describes the input-output relationship, $y = F(\mathbf{x})$
- Having available a finite set of experimental data, based on which a model (e.g. neural networks) can be built for simulation and prediction purposes

Due to the lack of an analytical model, one can only seek to determine the relative importance of the input variables through empirical methods. A complete analysis would require examination of all possibilities, e.g., taking two variables at a time to analyze their dependence or correlation, then taking three at a time, etc. This, however, is both infeasible (requiring 2^n experiments!) and not infallible (since the available data may be of poor quality in sampling the whole input space). In the following, therefore, we apply the technique of deleting one feature at a time [14] to rank the input features and identify the most important ones for intrusion detection using SVMs.

3.1 Methodology for Ranking Importance

We first describe the input ranking methodology: One input feature is deleted from the data at a time, the resultant data set is then used for the training and testing of the classifier. Then the classifier's performance is compared to that of the original classifier (based on all features) in terms of relevant performance criteria. Finally, the importance of the feature is ranked according to a set of rules based on the performance comparison. The procedure is summarized as follows:

1. Delete one input feature from the (training and testing) data
2. Use the resultant data set for training and testing the classifier
3. Analyze the results of the classifier, using the performance metrics
4. Rank the importance of the feature according to the rules
5. Repeat steps 1 to 4 for each of the input features

3.2 Performance Metrics

To rank the importance of the 41 features (of the DARPA data) in an SVM-based IDS, we consider three main performance criteria: overall accuracy of (5-class) classification; training time; and testing time. Each feature will be ranked as "important", "secondary", or "insignificant", according to the following rules that are applied to the result of performance comparison of the original 41-feature SVM and the 40-feature SVM:

Rule set:

1. **If accuracy decreases and training time increases and testing time decreases, then the feature is important**
2. **If accuracy decreases and training time increases and testing time increases, then the feature is important**
3. **If accuracy decreases and training time decreases and testing time increases, then the feature is important**
4. **If accuracy unchanged and training time increases and testing time increases, then the feature is important**
5. **If accuracy unchanged and training time decreases and testing time increases, then the feature is secondary**
6. **If accuracy unchanged and training time increases and testing time decreases, then the feature is secondary**
7. **If accuracy unchanged and training time decreases and testing time decreases, then the feature is insignificant**
8. **If accuracy increases and training time increases and testing time decreases, then the feature is secondary**
9. **If accuracy increases and training time decreases and testing time increases, then the feature is secondary**
10. **If accuracy increases and training time decreases and testing time decreases, then the feature is insignificant**

4. Experiments

Support vector machines are used for ranking the importance of the input features, taking above mentioned performance metrics and the rule set into consideration [15]. Once the importance of the input features was ranked, the classifiers were trained and tested with only the important features. Further, we validate our methodology by comparing the performance of the classifier using all input features to that using the important and the secondary features; and we also compare the performance of a classifier using the union of the important features for all five classes.

We note that because SVMs are only capable of binary classifications, we will need to employ five SVMs for the five-class identification problem in intrusion detection. But since the set of important features may differ from class to class, using five SVMs becomes an advantage rather than a hindrance, i.e., in building an IDS using five SVMs, each SVM can use only the important features for that class which it is responsible for making classifications.

The data set consists of 11982 randomly selected points, from the DARPA data set for training and testing. The training set consists of 5092 data points; where as the testing set consists of 6890 data points. The most important performance criterion, detection accuracy, is calculated as the percentage of correct classifications.

Accuracy = Number of correctly classified patterns / Total number of patterns

4.1 Support Vector Machines

Our results are summarized in the following tables. Table 1 gives the performance results of the five SVMs for each respective class of data. Table 2 through Table 6, each containing the results of 41 experiments; give the performance statistics of the SVM with 40 features. Table 7 shows the results of SVMs performing classification, with each SVM using as input the important features for all five classes. Table 8 shows the results of SVMs performing classification, with each SVM using as input the union of the important features for all five classes. Table 9 shows the result of SVMs performing classification, with each SVM using as input the important and secondary features for each respective class.

Table1: Performance of SVMs using 41 features

Class	Training Time (sec)	Testing Time (sec)	Accuracy
Normal	7.66	1.26	99.55%
Probe	49.13	2.10	99.70%
DOS	22.87	1.92	99.25
U2Su	3.38	1.05	99.87
R2L	11.54	1.02	99.78

Table2: Class 1, Normal

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	7.66	1.26	99.55
1.	10.19	1.11	99.51
2.	6.56	1.46	99.55
3.	9.06	1.47	99.48
4.	9.96	1.08	99.55
5.	33.11	1.62	99.19
6.	7.56	1.79	98.75
7.	7.11	1.43	99.55
8.	8.33	1.41	99.55
9.	8.37	1.37	99.55
10.	8.68	1.35	99.55
11.	7.49	1.33	99.55
12.	8.01	1.38	99.55
13.	7.14	0.81	99.55

14.	8.00	1.46	99.55
15.	9.81	1.43	99.55
16.	8.15	1.04	99.55
17.	8.12	1.47	99.55
18.	7.36	1.30	99.55
19.	8.00	1.12	99.55
20.	8.15	1.38	99.55
21.	7.98	1.42	99.55
22.	8.12	1.43	99.55
23.	7.65	1.34	99.56
24.	7.29	1.30	99.55
25.	8.32	1.35	99.55
26.	7.71	1.30	99.55
27.	7.73	1.38	99.55
28.	7.90	1.47	99.55
29.	7.81	1.39	99.55
30.	7.57	1.38	99.55
31.	7.11	1.30	99.55
32.	6.17	1.26	99.55
33.	8.53	1.51	99.48
34.	7.23	1.48	99.55
35.	6.96	1.35	99.55
36.	10.19	1.36	99.55
37.	6.74	1.33	99.55
38.	8.17	1.43	99.55
39.	7.75	1.32	99.55
40.	7.20	1.45	99.55
41.	9.38	1.43	99.55

Table3: Class 2, Probe

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	49.13	2.10	99.70
1.	58.93	2.01	99.70
2.	44.07	1.79	99.70
3.	51.00	2.19	99.61
4.	62.42	1.85	99.70
5.	75.67	1.97	98.14
6.	51.03	1.17	99.52
7.	51.62	1.98	99.70
8.	55.34	1.88	99.72
9.	53.05	1.99	99.70
10.	46.29	2.00	99.70
11.	45.68	1.96	99.70

12.	53.18	1.95	99.70
13.	55.27	1.95	99.70
14.	50.67	1.92	99.70
15.	49.50	2.07	99.70
16.	47.61	2.16	99.70
17.	49.38	1.93	99.70
18.	50.28	1.91	99.70
19.	50.33	1.94	99.70
20.	48.61	1.93	99.70
21.	50.40	1.89	99.70
22.	51.50	1.96	99.70
23.	49.00	2.63	99.46
24.	42.86	1.97	99.61
25.	52.40	1.95	99.71
26.	52.42	1.99	99.71
27.	62.51	2.05	99.71
28.	71.80	1.91	99.71
29.	45.95	1.78	99.70
30.	46.62	2.00	99.70
31.	46.35	1.93	99.70
32.	31.89	1.82	99.67
33.	50.90	1.83	99.62
34.	47.64	1.30	99.70
35.	49.49	1.87	99.70
36.	47.39	1.97	99.70
37.	48.19	2.03	99.70
38.	57.51	1.85	99.71
39.	52.54	1.94	99.71
40.	56.45	1.98	99.70
41.	51.66	1.71	99.70

Table4: Class 3, Denial of Service

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	22.87	1.92	99.25
1.	21.76	1.87	99.23
2.	23.60	1.89	99.25
3.	17.88	2.03	99.10
4.	20.00	1.79	99.25
5.	39.57	1.61	97.55
6.	19.63	0.84	98.07
7.	23.76	1.87	99.25
8.	31.23	1.86	99.20
9.	23.80	1.78	99.25
10.	27.01	1.82	99.25
11.	22.03	1.86	99.25
12.	19.69	1.84	99.25
13.	21.30	1.93	99.25
14.	20.18	2.02	99.25

15.	18.76	1.89	99.25
16.	21.56	1.78	99.25
17.	22.98	2.09	99.25
18.	21.47	1.95	99.25
19.	20.79	1.97	99.25
20.	21.49	1.96	99.25
21.	21.75	1.94	99.25
22.	24.93	2.01	99.25
23.	23.94	3.01	98.58
24.	25.43	2.05	99.20
25.	21.70	1.80	99.19
26.	25.93	1.98	99.19
27.	24.21	1.41	99.20
28.	26.16	1.80	99.20
29.	29.99	1.93	99.25
30.	18.27	1.79	99.20
31.	19.85	1.79	99.25
32.	11.70	0.95	98.69
33.	44.19	1.74	99.19
34.	28.27	1.88	99.25
35.	28.94	1.75	99.22
36.	27.39	1.80	99.22
37.	22.40	1.86	99.25
38.	22.45	1.95	99.19
39.	23.81	1.92	99.20
40.	50.15	1.84	99.22
41.	25.36	2.03	99.19

Table5: Class 4, User to Root

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	3.38	1.05	99.87
1.	2.98	0.96	99.87
2.	3.35	0.98	99.87
3.	3.00	1.04	99.87
4.	3.21	1.04	99.87
5.	3.11	0.65	99.72
6.	1.99	0.18	88.81
7.	3.40	1.07	99.87
8.	3.43	1.10	99.87
9.	3.37	0.97	99.87
10.	3.69	0.97	99.87
11.	3.47	1.06	99.87
12.	3.36	0.99	99.87
13.	3.61	1.01	99.87
14.	3.12	1.02	99.87
15.	3.40	1.11	99.87
16.	3.57	1.14	99.87

17.	3.39	0.98	99.87
18.	3.46	1.07	99.87
19.	3.41	1.05	99.87
20.	3.35	1.10	99.87
21.	3.34	1.08	99.87
22.	3.26	1.07	99.87
23.	3.39	1.05	99.87
24.	3.32	1.07	99.87
25.	3.44	1.09	99.87
26.	3.38	1.06	99.87
27.	3.36	1.05	99.87
28.	3.23	1.00	99.87
29.	3.36	0.98	99.87
30.	3.42	0.98	99.87
31.	3.34	1.00	99.87
32.	3.95	0.92	99.84
33.	4.58	0.99	99.85
34.	3.36	1.02	99.87
35.	2.98	1.05	99.87
36.	3.50	1.05	99.87
37.	3.43	1.00	99.87
38.	3.79	1.05	99.87
39.	3.27	1.07	99.87
40.	3.36	0.99	99.87
41.	3.36	1.01	99.87

Table6: Class 5, Remote to Local

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	11.54	1.02	99.78
1.	7.54	1.04	99.80
2.	8.79	1.23	99.78
3.	9.95	1.11	99.75
4.	8.56	1.26	99.78
5.	12.11	1.79	99.06
6.	16.52	0.63	98.88
7.	10.18	1.34	99.78
8.	9.59	1.31	99.78
9.	8.41	1.23	99.78
10.	9.30	1.32	99.78
11.	10.21	1.23	99.78
12.	9.48	1.33	99.78
13.	9.88	1.29	99.78
14.	8.84	1.22	99.78
15.	9.25	1.28	99.78
16.	8.89	1.20	99.78
17.	9.21	1.24	99.78
18.	9.60	1.30	99.78
19.	10.15	1.30	99.78

20.	10.68	0.99	99.78
21.	10.99	1.26	99.78
22.	10.88	1.26	99.78
23.	8.19	1.26	99.78
24.	7.67	1.22	99.72
25.	9.26	1.05	99.78
26.	10.11	1.30	99.78
27.	9.09	1.24	99.78
28.	9.10	1.23	99.78
29.	11.39	1.11	99.78
30.	10.64	1.26	99.78
31.	8.56	1.26	99.78
32.	11.55	1.05	99.80
33.	12.35	1.25	99.80
34.	10.59	1.14	99.78
35.	9.07	1.18	99.78
36.	9.22	1.22	99.78
37.	9.33	1.30	99.78
38.	8.98	0.95	99.78
39.	8.52	1.26	99.78
40.	8.98	1.11	99.78
41.	10.35	1.26	99.78

Table7: Performance of SVMs using important features

Class	No of Features	Training Time (sec)	Testing Time (sec)	Accuracy
Normal	25	9.36	1.07	99.59%
Probe	7	37.71	1.87	99.38%
DOS	19	22.79	1.84	99.22%
U2Su	8	2.56	0.85	99.87%
R2L	6	8.76	0.73	99.78%

Table8: Performance of SVMs using union of important features (30)

Class	Training Time (sec)	Testing Time (sec)	Accuracy
Normal	7.67	1.02	99.51%
Probe	44.38	2.07	99.67%
DOS	18.64	1.41	99.22%
U2Su	3.23	0.98	99.87%
R2L	9.81	1.01	99.78%

Table9: Performance of SVMs using important and secondary features

Class	No of Features	Training Time (sec)	Testing Time (sec)	Accuracy
Normal	39	8.15	1.22	99.59%
Probe	32	47.56	2.09	99.65%

DOS	32	19.72	2.11	99.25%
U2Su	25	2.72	0.92	99.87%
R2L	37	8.25	1.25	99.80%

5. Summary & Conclusions

Comparing Table 1 with Tables 7, 8, and 9, we observe that

- The most important features for the two classes of Normal and DOS heavily overlap
- U2Su and R2L, the two smallest classes representing the most serious attacks, each has a small number of important features and a large number of secondary features
- The performances of (a) using the important features for each class, Table 7, (b) using the union of important features, Table 8, and (c) using the union of important and secondary features for each class, do not show significant differences, and are all similar to that of using all 41 features
- Using the important features for each class gives the most remarkable performance: the testing time decreases in each class; the accuracy increases slightly for one class Normal, decreases slightly for two classes Probe and DoS, and remains the same for the two most serious attack classes.

Our ongoing experiments include making 23-class (22 specific attacks and normal) feature identification using SVMs and neural networks, for designing a cost-effective and real time intrusion detection tool.

We propose a five SVM based intrusion detection architecture, where we use the set of important features for each class that are responsible for making classifications.

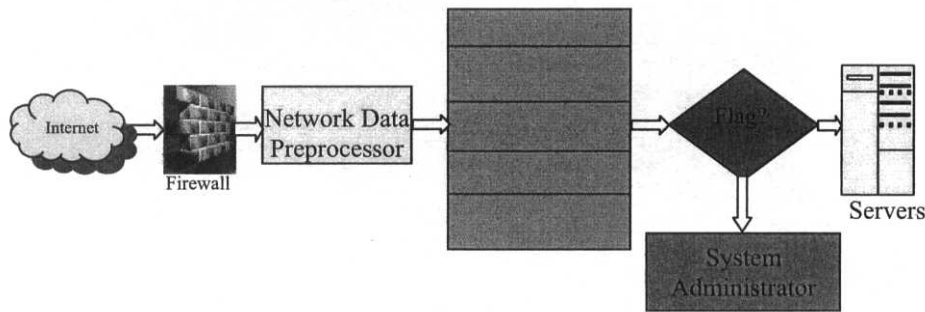


Figure1: Prototype of a SVM based IDS

6. Acknowledgements

Support for this research received from ICASA (Institute for Complex Additive Systems Analysis, a division of New Mexico Tech) and a U.S. Department of Defense IASP capacity building grant is gratefully acknowledged. We would also like to acknowledge many insightful conversations with Dr. Jean-Louis Lassez and David Duggan that helped clarify some of our ideas.

References

[1] Denning, D. (1987) An Intrusion-Detection Model. IEEE Transactions on Software Engineering, Vol.SE-13, No 2.

- [2] Kumar, S. & Spafford, E. H. (1994) An Application of Pattern Matching in Intrusion Detection. Technical Report CSD-TR-94-013. Purdue University.
- [3] Ghosh, A. K. (1999) Learning Program Behavior Profiles for Intrusion Detection. USENIX.
- [4] Cannady, J. (1998) Artificial Neural Networks for Misuse Detection. National Information Systems Security Conference.
- [5] Ryan, J., Lin, M-J. & Miikkulainen, R. (1998) Intrusion Detection with Neural Networks. Advances in Neural Information Processing Systems 10, MIT Press, Cambridge, MA.
- [6] Debar, H., Becke, M. & Siboni, D. (1992) A Neural Network Component for an Intrusion Detection System. Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy.
- [7] Debar, H. & Dorizzi, B. (1992) An Application of a Recurrent Network to an Intrusion Detection System. Proceedings of the International Joint Conference on Neural Networks, pp.78-83.
- [8] Luo, J. & Bridges, S. M. (2000) Mining Fuzzy Association Rules and Fuzzy Frequency Episodes for Intrusion Detection. International Journal of Intelligent Systems, John Wiley & Sons, pp.687-703.
- [9] Cramer, M., et. al. (1995) New Methods of Intrusion Detection using Control-Loop Measurement. Proceedings of the Technology in Information Security Conference (TISC-95), pp.1-10.
- [10] Mukkamala, S., Janowski, G. & Sung, A.H. (2001) Monitoring Information System Security. Proceedings of the 11th Annual Workshop on Information Technologies & Systems, pp.139-144.
- [11] Mukkamala, S., Janowski, G. & Sung, A.H. (2002) Intrusion Detection Using Neural Networks and Support Vector Machines. Proceedings of 2002 IEEE International Joint Conference on Neural Networks, pp.1702-1707.
- [12] Mukkamala, S. & Sung, A.H. (2002) Comparison of Neural Networks and Support Vector Machines in Intrusion Detection Workshop on Statistical and Machine Learning Techniques in Computer Intrusion Detection, <http://www.mts.jhu.edu/~cidwshop/abstracts.html>
- [13] <http://kdd.ics.uci.edu/databases/kddcup99/task.htm>.
- [14] Sung, A. H. (1998) Ranking Importance of Input Parameters Of Neural Networks. Expert Systems with Applications, pp.405-411.
- [15] Joachims, T. (2000) SVMlight is an implementation of Support Vector Machines (SVMs) in C. http://ais.gmd.de/~thorsten/svm_light. University of Dortmund. Collaborative Research Center on "Complexity Reduction in Multivariate Data" (SFB475).
- [16] Joachims, T. (1998) Making Large-Scale SVM Learning Practical. LS8-Report, University of Dortmund, LS VIII-Report.
- [17] Vapnik, V. N. (1995) The Nature of Statistical Learning Theory. Springer, Berlin, Heidelberg, New York.
- [18] Joachims, T. (2000) Estimating the Generalization Performance of a SVM Efficiently. Proceedings of the International Conference on Machine Learning, Morgan Kaufman.

This page intentionally left blank

Section 7

Features and Classification

This page intentionally left blank

A Trainable Classifier via k Nearest Neighbors

I. Mora-Jiménez, A. Lyhyaoui, J. Arenas-García

A. Navia-Vázquez and A.R. Figueiras-Vidal

Dept. of Signal Theory and Communications, UC3M, Spain

Abstract. This paper introduces a new classifier derived from a variant of the k -Nearest Neighbor (k NN) rule. This classification scheme, which we call k NN Learning Vector Classifier (k NN-LVC), has a similar architecture to that of Learning Vector Quantizers (LVQs). In fact, both methods place in the observation space a set of centroids or prototypes with a limited area of influence; however, our approach finds optimal prototypes by optimizing a new discriminant function that considers the k nearest prototypes to a sample. Among k NN-LVC characteristics are its localized nature, easy training and interpretation, small storage requirements, and a very competitive performance. The proposed technique is benchmarked against other classifiers as k NN and LVQ. Experiments show good generalization capabilities and efficacy of our approach on datasets with enough number of data in relation to dimensionality.

1 Introduction

A desirable property of any classifier, apart from being a reliable system, is that it provides an understanding of its operation. The k Nearest Neighbor (k NN) rule [1], [4], [5] fulfills this condition. It is a simple, understandable and very intuitive method for classification: the class membership of a sample is chosen to be the majority class among the k most similar patterns to the sample. Besides, this classifier presents an asymptotic error rate less than twice the Bayes error probability [3], what is quite interesting if we consider the k NN rule does not assume any statistical model for the underlying problem.

The main drawback of the k NN rule is that, to classify a sample, it needs to calculate the similarity between that sample and all the training patterns. Therefore, this classifier involves a heavy computational cost, apart from the need of storing all the training samples. In fact, these are the reasons why its use is not very extended, even though it provides very good recognition rates.

The Learning Vector Quantization (LVQ) scheme [7] proposed by Kohonen can be seen as an alternative to reduce the amount of stored patterns while controlling the error. LVQ is a supervised classification technique that partitions data space into regions, each one defined by a labelled prototype or centroid. During the learning phase, prototypes' locations are tuned in order to represent the probability density function (pdf) of the data. After training, LVQ classifies a sample with the label of its nearest prototype.

Our work is directed towards building a classifier having the good characteristics from both k NN and LVQ methods, whereas achieving better performance. To do so, we first start from a modified k NN scheme, which takes into account not only the class of the k nearest patterns but also a measure of their closeness. Then, we derive a discriminant function and apply

it to a pool of prototypes, obtaining a trainable version of the modified k NN rule. Optimization of this discriminant leads to the proposed k Nearest Neighbor Learning Vector Classifier (k NN-LVC). The main difference between LVQ and our scheme is that k NN-LVC does not pursue a model for the probability density function. Besides, the proposed classifier takes into account the closer prototype and the following runner-ups, what gives more potential to the classifier.

The outline of the paper is as follows. We first review the basics of the k NN rule and present our modified version. In Section 3 we derive k NN-LVC, together with a gradient based training scheme. Section 4 presents experimental results using several benchmarking problems and compare the performance of k NN-LVC with competitive classifiers as classical k NN and LVQs. Finally, we discuss the results and suggest further lines of work.

2 Classification via k Nearest Neighbors

Fundamentals of k NN rule [1], [4], [5] for classification come from the statistical approach to pattern recognition, where the purpose is to minimize the classification error.

One method for designing classifiers with the lowest error probability is the well-known Bayes' decision rule for minimum error [4]. Assuming \mathbf{x} is a pattern in an n -dimensional feature space, this rule assigns class C_i (i : class index) to \mathbf{x} if

$$P(C_i|\mathbf{x}) > P(C_j|\mathbf{x}) \quad \forall j \neq i \quad (1)$$

where $P(C_i|\mathbf{x})$ is the a posteriori probability of vector \mathbf{x} to belong to class C_i . Using Bayes theorem, $P(C_i|\mathbf{x})$ can be expressed as

$$P(C_i|\mathbf{x}) = \frac{P(C_i)p(\mathbf{x}|C_i)}{p(\mathbf{x})} \quad (2)$$

where $P(C_i)$ denotes the prior probability of class C_i , $p(\mathbf{x}|C_i)$ is the pdf of data belonging to C_i , and $p(\mathbf{x})$ corresponds to the unconditional data pdf.

Introducing eq.(2) into (1), and for a two-class case ($i = 0, 1$), we have the following classification rule: if

$$P(C_1)p(\mathbf{x}|C_1) > P(C_0)p(\mathbf{x}|C_0) \quad (3)$$

then \mathbf{x} is assigned to class C_1 ; otherwise class C_0 is selected. From now on, we will express this as

$$P(C_1)p(\mathbf{x}|C_1) \stackrel{c_1}{\geq} P(C_0)p(\mathbf{x}|C_0) \quad (4)$$

When dealing with a finite set of patterns with an unknown pdf, values in eq.(4) have to be substituted for their estimates. Assuming there are N_i samples for class C_i and N in total ($N = N_0 + N_1$), we can estimate the prior probabilities from the relative frequency

$$\hat{P}(C_i) = \frac{N_i}{N} \quad i = 0, 1 \quad (5)$$

As for the class conditional pdf, we propose to use the k Nearest Neighbor density estimator [1]. Let $d(\mathbf{x}, \mathbf{y})$ be the distance between \mathbf{x} and \mathbf{y} according to a certain metric d ; the k NN density estimation scheme first finds the k nearest neighbors to \mathbf{x} according to metric d ,

and then determines the minimum volume $v(\mathbf{x})$ centered in \mathbf{x} and containing these k samples. This way, the estimate is given by

$$\hat{p}(\mathbf{x}) \simeq \frac{k}{Nv(\mathbf{x})} \quad (6)$$

Using eq.(6) to approximate the class conditional density [1], we have

$$\hat{p}(\mathbf{x}|C_i) = \frac{k_i}{N_i v_{k_i}(\mathbf{x})} \quad i = 0, 1 \quad (7)$$

where k_i is the number of samples from C_i among the k nearest neighbors ($k = k_0 + k_1$). This formulation leads to the classical k NN rule, which assigns vector \mathbf{x} to the class with a higher k_i value. Optimal selection of parameter k depends on dataset size and dimensionality [1],[4],[5].

Fukunaga [5] proposes to proceed separately for each class, taking the k_1 nearest neighbors belonging to class C_1 and the corresponding k_0 from C_0 (with fixed values for k_0 and k_1), and then estimating the class-conditional densities as

$$\hat{p}(\mathbf{x}|C_i) = \frac{k_i}{N_i v_{k_i}(\mathbf{x})} \quad i = 0, 1 \quad (8)$$

where $v_{k_i}(\mathbf{x})$ is the volume centered in \mathbf{x} encompassing the k_i selected samples of class C_i .

Independent selection of k_0 and k_1 yields some advantage over classical k NN rule, although a fixed selection for the whole space can result in some disadvantage for certain regions. On the other hand, it should be noted that this flexibility implies additional computational cost, since it is necessary to explore values for two parameters instead of one.

To avoid this extra complexity while preserving the advantages inherent to Fukunaga's scheme, we propose to use an analogous procedure: fixing k , the total number of samples to consider, and extracting values k_i from it. This way, values k_1 and k_0 will vary depending on the sample to classify.

Now, from equations (4), (5) and (8), it is immediate to get the following classification rule

$$\frac{k_1}{v_{k_1}(\mathbf{x})} \stackrel{C_1}{\approx} \frac{k_0}{v_{k_0}(\mathbf{x})} \quad (9)$$

If we accept to use hyperellipsoidal volumes and, for simplicity, the same sample covariance matrix for both classes, eq.(9) can be expressed as

$$\frac{k_1}{d_{k_1}^n} \stackrel{C_1}{\approx} \frac{k_0}{d_{k_0}^n} \quad (10)$$

Assuming Euclidean distance (represented by $\|\cdot\|_2$) and spherical symmetry, we have the following version for the modified k NN rule

$$\frac{k_1}{\|\mathbf{x} - \mathbf{x}^{(k_1)}\|_2^n} \stackrel{C_1}{\approx} \frac{k_0}{\|\mathbf{x} - \mathbf{x}^{(k_0)}\|_2^n} \quad (11)$$

where $\mathbf{x}^{(k_i)}$ is the furthest neighbor belonging to class C_i and n is the dimensionality of the data.

The main property of our k NN version is that it takes advantage of the information about the distance to the k_i -th nearest sample of C_i , but it needs just one parameter to build the classifier. Nevertheless, it still demands large memory, as well as a substantial computational cost. To overcome these difficulties we devise the k NN Learning Vector Classifier, which is presented in the next section.

3 The proposed k NN Learning Vector Classifier (k NN-LVC)

After slight changes, the k NN version proposed in Section 2 is appropriate to design local classifiers with low storage requirements, in addition to allow gradient trainable schemes. It suffices to choose a pool of centroids $\{c_0\}$ and $\{c_1\}$ for classes C_0 and C_1 , respectively, and introduce them in eq.(11). After some rearrangement we obtain

$$\frac{\|x - c_0^{(k_0)}\|_2^2}{k_0^{2/n}} - \frac{\|x - c_1^{(k_1)}\|_2^2}{k_1^{2/n}} \stackrel{c_1}{\geq} \stackrel{c_0}{\leq} 0 \quad (12)$$

Assuming t is the target value for sample x ($t = 1$ if $x \in C_1$, $t = -1$ if $x \in C_0$), we derive from (12) the following discriminant function D

$$D = t \left(\frac{\|x - c_0^{(k_0)}\|_2^2}{k_0^{2/n}} - \frac{\|x - c_1^{(k_1)}\|_2^2}{k_1^{2/n}} \right) \quad (13)$$

It is immediate to check that a sample x is correctly classified by this method if $D > 0$. Therefore, our learning algorithm will modify progressively the centroids' positions to find a maximum of D . To do so, we apply the method of steepest gradient ascent. Taking the gradients of D with respect to the k_1 -th and k_0 -th prototypes we have

$$\nabla_{c_1^{(k_1)}} D = \frac{2t}{k_1^{2/n}} (x - c_1^{(k_1)}) \quad (14a)$$

$$\nabla_{c_0^{(k_0)}} D = -\frac{2t}{k_0^{2/n}} (x - c_0^{(k_0)}) \quad (14b)$$

In practice, we apply gradient ascent to all the $k = k_0 + k_1$ nearest centroids. So, when a training sample x is taken, we use the following updating rules for its k nearest prototypes

$$c_1^{(m_1)}(l+1) = c_1^{(m_1)}(l) + \alpha \frac{2t}{m_1^{2/n}} (x - c_1^{(m_1)}(l)), \quad 1 \leq m_1 \leq k_1 \quad (15a)$$

$$c_0^{(m_0)}(l+1) = c_0^{(m_0)}(l) - \alpha \frac{2t}{m_0^{2/n}} (x - c_0^{(m_0)}(l)), \quad 1 \leq m_0 \leq k_0 \quad (15b)$$

where α stands for the learning rate and l represents the training iteration.

The k NN-LVC training phase starts with a set of labelled centroids. Then, the learning proceeds by sequentially taking training patterns and using equations (15a,b) to update the locations of the k nearest centroids to each pattern.

Note that the effective learning rate ($\alpha_e = 2\alpha/m_i^{2/n}$) is different for every centroid: the furthest prototype has the lowest α_e , what is used to compensate for the longest displacement vector ($x - c_i^{(k_i)}$).

Regarding selection of parameter k , it is made through an exploration process depending on factors like dimensionality and number of prototypes (as in the k NN methods). An additional aspect to consider when using a reduced number of centroids is that we should not use very high values for k , in order to keep fundamentals of the k NN procedures as local classifiers¹.

4 Experimental work

In this section we will check k NN-LVC performance by comparing it to other classifiers with a similar structure, including both trainable and non-trainable approaches. Let us begin with an outline of these classification schemes:

- k NN: labels a sample according to the majority class among its k nearest neighbors from the whole training set.
- Reduced k NN: is a k NN classifier applied to a reduced set of training samples, randomly selected among the training set. It involves lower computational and storage requirements in comparison to the previous classifier.
- LVQ1: is the simplest algorithm of the LVQ type. It also starts from a reduced set of prototypes, although this is an iterative learning method. In every learning stage, LVQ1 chooses a training sample and determines its closest prototype. This prototype is moved towards the sample if they belong to the same class; otherwise, it is moved away from the pattern. In both cases the displacement is proportional to the distance between the sample and the prototype. This proportionality factor α is the learning step for LVQ1. As for classification, LVQ1 assigns a sample to the same class as that of its nearest prototype.
- LVQ3 is an improved version of LVQ1, where the two nearest prototypes (say c_1 and c_2) to a sample x are simultaneously updated in the following manner:
 - if x , c_1 and c_2 belong to the same class, these two prototypes are moved towards x a multiple ϵ of the learning rate α .
 - if just c_1 or c_2 has the same label as x and the sample falls into a zone, called window (w), around the midplane of c_1 and c_2 , LVQ3 shifts the prototypes towards or away from x , depending on whether prototype and sample have the same label or not, respectively.

Note that this modification of LVQ1 is largely heuristic, in addition to require two additional parameters (ϵ and w) for the learning phase. On the other hand, classification with LVQ3 is identical to that of LVQ1.

It is interesting to remark that the proposed k NN-LVC can be seen as a generalization of the previous trainable approaches. So then, LVQ1 is equivalent to k NN-LVC when $k = 1$. Regarding LVQ3, we could consider that its training is a variant of the k NN-LVC learning when $k = 2$ and some restrictions are added. Nevertheless, the goal is different: LVQ-type

¹We consider techniques such as k NN to be local in the sense that the assignment of a sample to a class only depends on a near area around the sample, no matter what happens in further regions.

methods seek to place prototypes in order to approximate the class distributions at the same time as achieving good accuracy rates, while k NN-LVC is only concerned about the latter.

As for parameters used by the previous classifiers, we have explored different values for all of them, reporting the best results in our experiments. We indicate below the range of values we have considered for each parameter:

- number of prototypes: except for k NN, where all training samples are considered as prototypes, we have tested some values between 5% and 20% of the training set. However, when the best accuracy was achieved with 5% of the centroids, we further decreased this percentage until performance worsened. All methods start with the same prototypes, which are randomly chosen among the samples of each class.
- parameter k is only used in the case of k NN rules and k NN-LVC. Exploration range for k goes from 1 to the number of prototypes minus one.
- learning rate α : as suggested in [7], we have initialized it large enough to achieve fast adjustment and have gradually decreased it for fine-tuning. In our experiments we have used a linearly decreasing towards zero schedule, with 10 different initial values within $[0.005, 0.6]$.
- number of epochs: we have sequentially presented all training samples 60 times, having checked this is enough for all the methods to converge.
- regarding additional parameters of LVQ3, we have explored the ranges recommended in [7], testing three values for each one: $w = \{0.2, 0.25, 0.3\}$ and $\epsilon = \{0.1, 0.25, 0.5\}$.

The comparison of performance for these classification schemes will include the accuracy rate, standard deviation and computational burden. We will consider two different components for the last one: searching of the nearest prototypes and their tuning (which only concerns the training stage).

We have applied these classification techniques to five pattern recognition problems extensively used as benchmarks. The first one, denominated “kwok”, is a synthetic 2-dimensional dataset proposed in [8], with 500 – 10200 (train–test) samples and having a 88.7% Bayesian classification limit. The rest are from the UCI Machine Learning Repository [2]: Waveform Data Generator (“waveform” in this paper) has 21 features and 4000 – 1000 patterns; Image Segmentation (“image”) is a 18 dimensional dataset with 1848 train and 462 test samples; Abalone (“abalone”) has 8 dimensions and 2507 – 1670 samples; Pima Indians Diabetes (“diabetes”) is a 8-dimensional dataset (768 samples) with no train–test partition, so in this case we have created 10 different partitions with a 60% – 40% train–test percentage, preserving in all partitions the prior probabilities. Excepting “waveform” and “diabetes”, where approximately $1/3$ of samples belong to C_1 , the rest of problems have a quite similar number of patterns for both classes. In “waveform”, “image” and “abalone” we have used the common binary version of the multiclass problem. We have preprocessed data so that all features have values within $[-1, 1]$. We should remark that all these problems have more than fifty training samples per dimension, what we think is enough so that we can apply k NN-type techniques.

Results in Table 1 show the average accuracy (in %) and standard deviation (in brackets) on the test set, obtained through 10 runs with different initial prototypes. The corresponding parameters are displayed in Table 2. Except for the k NN rule, the first value in Table 2

indicates the percentage of the training set used as prototypes; parameter k in the first two columns and k NN-LVC denotes the number of nearest prototypes used to classify a sample; α is the initial value for the learning step in the trainable methods; finally, we also show parameters w and ϵ for LVQ3.

Table 1: Average classification accuracy (in %) and standard deviation (in brackets) for different classification methods. Results in boldface correspond to the best accuracy for each problem

	k NN	Reduced k NN	LVQ1	LVQ3	k NN-LVC
kwok	88.0	86.9 (0.8)	87.8 (0.1)	87.8 (0.4)	88.0 (0.1)
waveform	92.1	89.3 (1)	91.7 (0.4)	92.0 (0.2)	91.9 (0.4)
image	97.4	92.9 (1)	96.1 (0.5)	96.3 (0.5)	96.4 (0.4)
abalone	79.5	76.7 (1)	78.8 (0.4)	79.2 (0.5)	79.7 (0.4)
diabetes	74.5	71.0 (1.9)	76.5 (3.1)	76.2 (3.1)	77.0 (1.8)

Table 2: Parameters for the classification methods and benchmark problems

	k NN	Reduced k NN	LVQ1	LVQ3	k NN-LVC
kwok	$k = 64$	20% $k = 13$	2.5% $\alpha = 0.6$	2.5%, $\alpha = 0.005$ $w = 0.25, \epsilon = 0.1$	10%, $k = 5$ $\alpha = 0.1$
waveform	$k = 57$	20% $k = 35$	0.4% $\alpha = 0.1$	0.4%, $\alpha = 0.005$ $w = 0.25, \epsilon = 0.25$	1.1%, $k = 3$ $\alpha = 0.01$
image	$k = 1$	20% $k = 1$	20% $\alpha = 0.3$	8%, $\alpha = 0.02$ $w = 0.3, \epsilon = 0.25$	20%, $k = 3$ $\alpha = 0.1$
abalone	$k = 39$	20% $k = 13$	4% $\alpha = 0.2$	4%, $\alpha = 0.02$ $w = 0.3, \epsilon = 0.1$	20%, $k = 9$ $\alpha = 0.02$
diabetes	$k = 29$	20% $k = 15$	3% $\alpha = 0.3$	2%, $\alpha = 0.3$ $w = 0.25, \epsilon = 0.25$	6.5%, $k = 3$ $\alpha = 0.1$

From these experiments, we observe that k NN and k NN-LVC get the best recognition rates. Both classifiers provide comparable results in all problems excepting “image” and “diabetes”. In “image”, the best k NN result is achieved considering only the nearest prototype, so it seems reasonable that reducing the number of prototypes and considering more than the nearest one will not lead to better accuracy. The case of “diabetes” is the opposite: k NN-LVC beats k NN, and this is because parameter k in k NN is quite high in relation to the dataset size. This fact allows that, even though the number of prototypes decrease considerably, it is possible to make a reliable estimate of the class membership considering more than the nearest centroid.

It is important to note that the classifier we propose requires less than 20% of the k NN prototypes, what implies k NN classification and storage costs are very high in comparison to k NN-LVC. Furthermore, the number k of prototypes used to classify a sample is much lower in our approach, what is also advantageous to reduce computational cost. Among all the classifiers compared in this section, the k NN scheme is the more demanding with respect to the searching phase, since it has the higher number of prototypes (the whole training set).

When the k NN method is applied over a reduced dataset (see ‘Reduced k NN’ in Table 1), it leads to a worsening in the recognition rates, suggesting that a learning phase for the prototypes’ location is quite advisable. In this case we could make the same discussion as in the previous classifier concerning the computational load, taking into account that the training set has been reduced.

As for the trainable schemes, we observe that k NN-LVC always outperforms LVQ1, probably because our method takes into account more than the nearest prototype. When we compare k NN-LVC with LVQ3, our approach usually achieves better recognition rates too. Regarding the computational load of k NN-LVC and LVQ-type approaches, our classifier requires more centroids (see parameters in Table 2) and so a relatively higher searching time. We should remark here that this additional cost in the search can be reduced if we apply techniques as the proposed in [6]. As far as the centroids’ tuning is concerned, the cost depends on the number of prototypes to update, as well as on the number of iterations. As the last parameter is the same for all the trainable methods, they just differ in the number of prototypes to update (k): it is limited to $k = 1$ for LVQ1 and $k = 2$ for LVQ3; k NN-LVC however explores different k values and, to this respect, our classifier requires some additional computation. Nevertheless, from the accuracy rates we notice the relevance of this parameter in the design of the classifier.

Apart from that, the standard deviations (Table 1) of k NN-LVC are usually lower than those of the other schemes, indicating that our scheme offers a more robust behaviour. Additionally, the design of k NN-LVC is simpler than that of LVQ3, since our approach involves less parameters to adjust.

From our experiments, we have noticed that the use of many centroids in LVQ-type methods can produce unwanted generalization effects. This is specially a problem when overlap is high, since there will be prototypes near the boundary that will produce misclassifications. We have checked that k NN-LVC reduces this effect since the training samples in the overlap region push the centroids away from the boundary area, thus providing better generalization.

In Figure 1 we illustrate the evolution of k NN-LVC accuracy when the proposed technique is trained with the parameters of Table 2 and different values for k . We observe that the general behavior (except for “image”) is an initial increase in accuracy as we consider more prototypes, until their number is too high and the method loses its localized fundamentals. This explains the usual higher accuracy of k NN-LVC in comparison to LVQ-type techniques.

5 Conclusions and further work

In this paper we have presented the k Nearest Neighbor Learning Vector Classifier (k NN-LVC), a trainable scheme derived from a new variant of the k NN technique and the Bayes’ decision rule for minimum error. We have shown that k NN-LVC is a simple method which preserves the fundamentals of k NN classifiers while solving the main problem of the k NN rule: its heavy storage and computational burden.

Quite similar to k NN-LVC as for architecture, interpretability and operation are the LVQ-type methods. In fact, we have shown that k NN-LVC can be considered as a generalization of LVQ when we remove the constraint of modelling the data distribution. Nevertheless, the k NN-LVC’s capacity to take into account k prototypes provides our method with more potential.

Experimental results through five benchmark problems have demonstrated the good per-

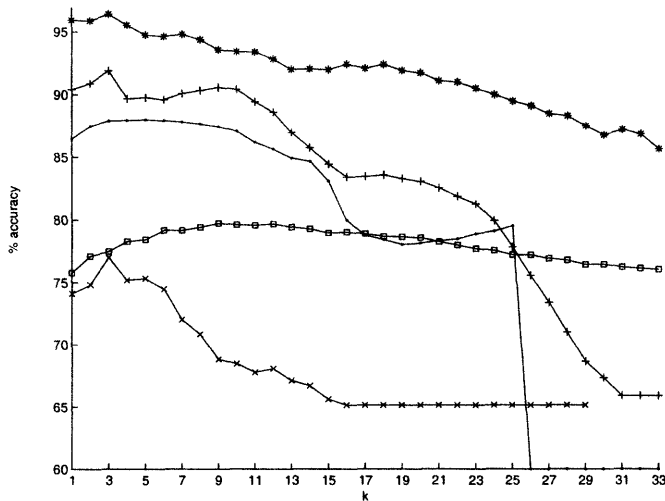


Figure 1: Evolution of the average accuracy on the test set as a function of parameter k for the k NN-LVC method. Represented problems are: “kwok” (dot marker), “waveform” (plus marker), “image” (star marker), “abalone” (square marker) and “diabetes” (cross marker).

formance of k NN-LVC in cases with enough number of training data in relation to dimensionality, both in comparison to k NN and LVQ techniques.

Regarding applications, we could further exploit the k NN-LVC gradient trainable structure and use it in non-static problems, leading to adaptive schemes of great interest in settings as speech processing or communication channels.

Ongoing research includes application of k NN-LVC in schemes with many initial prototypes (similar to dense clustering), as well as dealing with the outputs of ensembles of classifiers.

References

- [1] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK (1995).
- [2] C.L. Blake and C.J. Merz, *UCI Repository of machine learning databases*, University of California, Irvine, Dept. of Information and Computer Sciences, [<http://www.ics.uci.edu/~mllearn/MLRepository.html>] (1998).
- [3] T. Cover and P. Hart, Nearest Neighbor Pattern Classification, *IEEE Transactions on Information Theory* **13** (1967) 21–27.
- [4] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons (1973).
- [5] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, 2nd edn., New York (1990).
- [6] K. Fukunaga and P. Narendra, A branch and bound algorithm for computing k -nearest neighbors, *IEEE Transactions on Computers* **24** (1975) 750–753.
- [7] T. Kohonen, The Self-Organizing MAP, *Proc. IEEE* **78** (1990) 1464–1480.
- [8] J.T. Kwok, Moderating the Output of Support Vector Machine Classifiers, *IEEE Transactions on Neural Networks* **10** (1999) 1018–1031.

Combining Classifiers with Multimethod Approach

Mitja Lenič, Peter Kokol

*Laboratory for system design, Faculty of electrical engineering and computer science,
University of Maribor, Maribor, Slovenia
{mitja.lenic, kokol}@uni-mb.si*

ABSTRACT

The automatic induction of classifiers from examples is an important technique used in data mining. One of the problems encountered is how to induce a *good* classifier without overfitting. Although there is a lot of research going on in this field, the research is mainly focused on a specific machine learning method or on a specific combination of those methods. In this paper a multimethod approach to combine classifiers is presented that combines advantages of single methods and avoids their disadvantages at the same time by applying different methods on the same knowledge model, each of which may contain inherent limitations, with the expectation that the combined multiple methods may produce better results.

1. Introduction

The growing commercial interest in knowledge discovery and data mining techniques has stimulated new interest in the automatic induction of classifiers from examples. The research field has long tradition in knowledge extraction which is obviously needed in today's society that has relatively inexpensive and generally available means to collect and store the data from various environments. The huge amount of collected data causes great difficulties to extract useful information for decision support, discovery of underlying principles and other data analysis tasks. The variety of environments makes it almost impossible to develop single induction algorithm that would fit all possible requirements. Therefore the researchers have focused on different approaches to cope with presented challenge.

The question how the learning capacity can be increased and generalized has received much attention in particular in machine learning community [1]. The main problem is to learn how to learn and to be able to present new knowledge in the form easy understandable to the human.

Most of the work has concentrated on single approach with single knowledge representation. Some of the approaches are described briefly in Section 2. In recent years the explosion of hybrid methods can be observed. One of the possible reason is that separate research communities on symbolic machine learning, computational learning

theory, neural networks, statistics and pattern recognition, etc. have discovered each other. The brief overview of the idea of hybrid approaches is given in Section 3. Then a new technique for combining multiple methods is introduced in Section 4. In section 5 a description of some real-world problems and experimental results are presented.

Main contributions of our research are:

- an environment enabling to “play” with classical single or hybrid approaches and concepts
- an innovative multimethod approach enabling not only to combine single machine learning approaches but to integrate different approaches in on model, i.e. construction of various parts of a single decision tree with different methods
- standardisation of knowledge representations.

2. Single method approaches

Machine learning community has a long tradition in knowledge extraction that can be traced at least as far as the mid-1960. Trough the time different approaches evolved, such as symbolic approaches, computational learning theory, neural networks, etc. Most of the strength was and is concentrated in finding a way to extract generalized knowledge from specified examples. They all use inductive inference, that is the process of moving from concrete examples to general model(s), where the goal is to learn how to extract knowledge from objects by analyzing a set of instances (already solved cases) whose classes are known. Instances are typically represented as attribute-value vectors. Learning input consists of a set of vectors/instances, each belonging to a known class, and the output consists of a mapping from attribute values to classes. This mapping should accurately classify both the learning instances and also new unseen instances.

If we would compare those approaches there is no clear winner. Each method has some advantages and some inherent limitations compared to the other.

2.1 Decision trees

A decision tree [7] is a formalism for expressing the mapping from attribute values to classes and consists of tests or attribute nodes linked to two or more subtrees and leafs or decision nodes labeled with a class which represents the decision. A test node computes outcome based on the attribute values of an instance, where each possible outcome is associated with one of the subtrees. Most of the work is concentrated in top-down induction of decision trees (TDIDT) algorithm using different purity measures for the node splits. The principal problem with TDIDT and other algorithms is overfitting. Beyond a certain point specializing a tree by adding further splits can become counter-productive. That problem can be found not only in decision tree based classifier but also in other induction algorithms. To avoid the overfitting of decision trees the different strategies of pruning were introduced.

The greatest advantage of decision tree is that the knowledge representation is easily understandable to the humans and can be used without the computer.

2.2 Cognitive approaches

Another approach to knowledge extraction is to simulate cognitive abilities of the human brain. The most common representative of cognitive approach is artificial neural network. This approach has been verified by the nature and has ability to learn from examples. The main difference to the decision trees is in knowledge representation. The neural network stores the knowledge in the weights of connections to the neurons. That representation is not easy understandable to the human, but has very good characteristics for identification of complex patterns in learning data. The big question in using the cognitive approaches is to select right topology for the problem. If the topology is not appropriate the model might overfit or forget already learned instances.

2.3 Genetic algorithms

Genetic algorithms (GA) are adaptive heuristic search methods that may be used to solve all kinds of complex search and optimization problems [8], for which no efficient heuristic method has been developed. They are based on the evolutionary ideas of natural selection and genetic processes of biological organisms. Simulating the principles of natural selection and “survival of the fittest” first laid down by Charles Darwin genetic algorithms are able to evolve solutions to real-world problems. They are often capable of finding optimal solutions even in the most complex search spaces or at least they offer significant benefits compared to other search and optimization techniques.

Extraction of knowledge is a complex task, but in many cases an exact heuristic method exists, that usually works efficiently and reliably [7]. At the first glance there is no reason to use genetic algorithms. Nevertheless, there are some objective reasons that justify evolutionary approach. First, genetic algorithms provide a very general concept, that can be used in all kinds of problems. Because of their robustness they perform better on incomplete, noisy data and the problems which cannot be successfully solved by traditional techniques. Furthermore, genetic algorithms are not inherently limited to suboptimal solutions, because they use evolutionary operations such as mutation, recombination and selection, and can find solutions that can be easily overlooked otherwise [10]. Another important advantage of the evolutionary approach is, that not only one, but several equally qualitative solutions are obtained for the same problem. In this way the same decision can be made based on different features and/or combination of the features, which are essential to identify wrong decisions in one of many constructed solutions. An expert can decide which of the given solutions can be used. By weighting different parameters, searching can be directed to the solution that best applies to current needs, particularly in multi-class decision making processes.

There are many other approaches, like representation of the knowledge with rules, raw-sets, case based reasoning, support vector machine, different fuzzy methodologies, ensemble methods [2], but all have the common problem: How to find optimal solution i.e. learning how to learn.

3. Hybrid approaches

The hybrid approaches rest on the assumption that only in the synergetic combination of single models can unleash their full power. Each of the single method has its advantages but also inherent limitations and disadvantages, which must be taken into account when using the particular method. For example: symbolic methods usually represent the knowledge in

human readable form, the connectivists methods perform better in classification of unseen objects and are not so affected to the noise as symbolic methods. Therefore the logical step is to combine different methods to overcome the disadvantages and limitations of a single method.

In general the hybrids that combine single method can be divided according to the control flow into four categories [4]:

- sequential hybrid (chain-processing) – the output of one method is an input to another method. For example, the neural net is trained with the training set to reduce noise. Then the neural net and the training set are used to generate decision tree with the neural net decision's.
- parallel hybrid (co-processing) – different methods are used to extract knowledge. In the next phase some arbitration mechanism should be used to generate appropriate results
- external hybrid (meta processing) - one method uses another external one. For example meta decision trees [5], that uses neural nets in decision nodes to improve the classification results
- embedded hybrid (sub-processing) – one methods is embedded in another. That is the most powerful hybrid, but the least modular one, because usually the methods are tightly coupled.

The hybrid systems are commonly static in the structure and cannot change the order or sequence of application of single method.

To be able to use embedded hybrids of different internal knowledge representation, it is commonly required to transform one method representation into another. Some transformations are trivial, especially when converting from symbolic approaches. The problem is when the knowledge is not so clearly presented like in a case of the neural network [3][12]. That issue is also very important in our multimethod approach.

4. Multimethod approach

While studying presented approaches we were inspired by the idea of hybrid approaches and evolutionary algorithms. Both approaches are very promising in achieving the goal to improve the quality of knowledge extraction and are not inherently limited to suboptimal solutions. We also noticed that almost all attempts to combine different methods use loose coupling approach. The methods work almost independent of each other and therefore a lot of luck is needed to make them work as a team.

Each of those methods uses its own internal knowledge representation (symbolic, conectivistic) that other methods cannot reuse, because of the incompatibility of knowledge representation. That presents a major obstacle when trying to combine different methods, because knowledge is not exchangeable.

As already mentioned the idea of knowledge exchange is not new. There has been a lot of research going on in the extraction of knowledge from neural nets and vice versa. Although conversions are not ideal the majority of knowledge can be transformed from one to another. That gives us the possibility to exchange knowledge between different methods, and with proper application of those different methods hopefully significant improvements can be achieved. The idea is to dynamically combine and apply different methods in not predefined order to the same problem or the decomposition of the problem .

Another aspect of knowledge exchange is that methods have to be able to accept already constructed knowledge representation and have to apply its operations with the objective to improve the quality of the extracted knowledge. They have to be able to construct a knowledge representation from the scratch.

4.1. Combining the methods

Main concern of the mutlimethod approach is to find a way to enable dynamic combination of methodologies to the somehow quasi unified knowledge representation. Multiple equally qualitative solutions like in GA approach, where each solutions is gained using application of different methodologies with different parameters was used. Therefore we introduced a population composed out of individuals/solutions that have the common goal to improve their classification abilities on a given environment/problem. We have also enabled coexistence of symbolic and cognitive representation in the same population. The most common knowledge representation models have to be standardized to support the use different methods on individuals. In that manner the transformation support between each individual method does not need to be provided. The action is based on the assumption that it is highly improbable to find unified representation for all knowledge representations, therefore we decided to standardize the most popular representations like neural nets, decision trees, rules, etc. Standardization brings in general greater modularity and interchangeability, but it has following disadvantages - already existing methods cannot be directly integrated and have to be adjusted to the standardized representation.

Usually methods are composed out of operations that can be reused in other methods. Therefore we introduced the operation on an individual, a function that transforms one or more individuals to a single individual. Operation can be part of one or more methods, like pruning operator, boosting operator, etc.

The transition between knowledge representation is also introduced on the individual operator level, therefore the transition from one knowledge representation to another is presented as a method. Operator based view provides us with the ability to simply add new operations to the framework (figure 1).

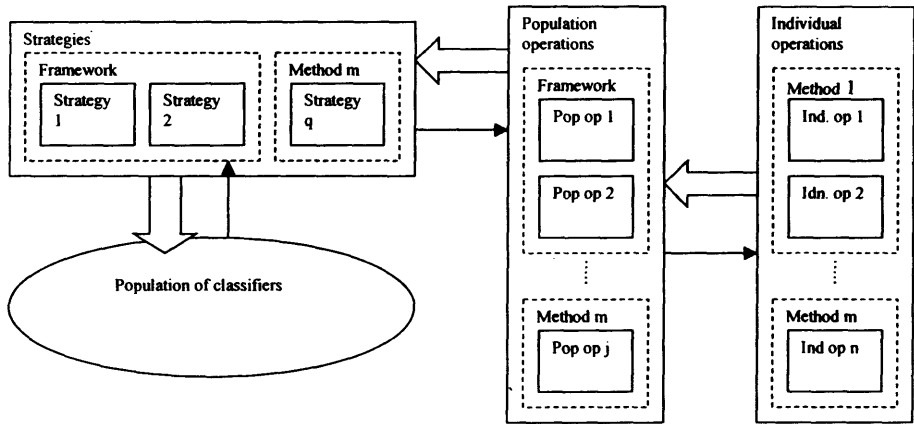


Figure 1: Multimethod framework

In general the representation with individual operations facilitate an effective and modular way to represent the result as single individual, but in general the result of

operation can be a population of individuals (for example mutation operation in GA is defined on individual level and on the population level). Therefore population operations that generally accept a population as the input and return the population as the result were introduced. The single method itself is composed out of population operations that use individual operations and is introduced as a strategy in the framework that improves individuals in a population (figure 1). Population operators can be generalized with higher order function and thereby reused in different methods.

To increase the modularity and extensibility of the framework the idea of object oriented paradigm has been used. The polymorphism and inheritance in operations and individual representations has been introduced. We extended the idea with aspect oriented paradigm, that enables clear separation of concerns and avoids tangled individual representation.

With this approach we achieved the modularity and extensibility of the framework without a lot of constraints to the implementation of methods. The individual and population operations, can be easily waved together with no additional effort.

The modular design of the framework enables us to apply some subparts of a method to another method. For example, independent of the induction method used, universal pruning and error reducing operations can be applied on a decision tree. The aspects of individual representation and methods/operations on the individual are strictly separated. The strategy can store some information inside the individual, that is not accessible to another method. Each method can only rely on the standardized part.

4.2. Monitoring the progress

The main concern of the multimethod framework is how to provide the meta level services to manage the available resources and the application of the methods. We extended the quest for knowledge into another dimension that is the quest for the best application order of methods. The problem that arises is how to control the quality of resulting individuals and how to intervene in case of not so good results. Because of different knowledge representation, the solutions cannot be trivially compared to each other and the assessment of which method is better is hard to imagine. Individuals in a population cannot be explicitly evaluated and the explicit fitness function cannot be easily calculated. Therefore the comparison of individuals and the assessment of the quality of the whole population cannot be given. Even if some criteria to calculate fitness function could be found, it would be probably be very time consuming and computational intensive. Therefore the idea of classical evolutionary algorithms controlling the quality of population and guidance to the objective cannot be applied.

Almost all methods require some parameters that affect their operations and the way of knowledge extraction. By using different parameters, the generalization ability can dramatically increase. The meta multimethod level is self adapting and does not require parameters from the user.

To achieve self-adaptive behavior of the evolutionary algorithm the strategy parameters have to be coded directly into the chromosome [1]. But in our approach the meta level strategy does not know about the structure of the chromosome, and not all of the methods use the EA approach to produce solution. Therefore for meta level chromosomes the parameters of the method and its individuals are taken. When dealing with self-adapting population with no explicit evaluation/fitness function there is also an issue of the best or most promising individual [9]. But of course the question how to control the population size or increase selection pressure, must be answered effectively.

To overcome above problem we have introduced three basic types of operations on the population: operations for reducing the population size, operations for maintaining the population size and operations for increasing the population size. These operations can be introduced by the method (i.e. evolutionary induced classifier) or are already integrated in the multimethod framework. The meta level strategy can use different operators to improve selection pressure and control population size. Population size is determined on the available computer resources.

4.3. Current status

The multimethod framework is implemented in Java. Currently we are adapting existing methods to work inside the framework and unifying the knowledge representation inside a single methodology. For the individual representation the notion of polymorphism is introduced. The framework operations and method operations are designed in the way that can accept different (sub)types of the individuals. That enables us to introduce generalized operators, that can be executed on generalized and specialized knowledge representation.

Currently we are standardizing internal representation for cognitive methods and are implementing heuristics and evolutionary approaches for vector decision induction [9]. With evolutionary approaches we attend to find generalized knowledge specialized by the heuristic methods and thereby avoid local optimum. With incremental addition we also validated the modular design of the multimethod framework, that is a good test bed for a new methods.

5. Experimental results

To make objective assessment of our method a comparison of obtained classifiers was made with reference methods C4.5, C5/See5 without boosting, C5/See5 with boosting, and genetic algorithm for decision tree construction [10]. The following quantitative measure were used:

$$accuracy = \frac{\text{num of correctly classified objects}}{\text{num. of all objects}}$$

$$accuracy_c = \frac{\text{num of correctly classified objects in class } c}{\text{num. of all objects in class } c}$$

$$\text{average class accuracy} = \frac{\sum_i accuracy_i}{\text{num. of classes}}$$

We decided to use average class accuracy instead of sensitivity and specificity, that are usually used when dealing with medical databases. Experiments have been made with seven real-world databases from the field of medicine.

5.1 MVP database

Prolapse is defined as the displacement of a bodily part from its normal position. The term mitral valve prolapse (MVP), therefore, implies that the mitral leaflets are displaced relative

to some structure, generally taken to be the mitral annulus. The silent prolapse is the prolapse which can not be heard with the auscultation diagnosis and is especially hard to diagnose. The implications of the MVP are the following: disturbed normal laminar blood flow, turbulence of the blood flow, injury of the chordae tendinae, the possibility of thrombus' composition, bacterial endocarditis and finally hemodynamic changes defined as mitral insufficiency and mitral regurgitation. Mitral valve prolapse is one of the most prevalent cardiac conditions, which may affect up to five to ten percent of normal population and one of the most controversial one.

Using the Monte Carlo sampling method 900 children and adolescents were selected representing the whole population under eighteen years of life. Routinely they were called for an echocardiography no matter of prior findings. 631 of them passed an examination of their health state in a form of a carefully prepared protocol specially made for the syndrome of MVP. The protocol consisted of 103 parameters that can possibly indicate the presence of MVP. Distribution of the three decision classes were: 5% prolapse, 6% silent prolapse, and 89% ok.

5.2 MRACID database

Through the help of classical medical research it has been established that surgeries under the general anesthesia cause in organism a tendency to dropping the blood's pH value, also known as predisposition to acidemia. The results of blood's gases analysis, serum electrolytes analysis, blood count and values of length of the operation, blood pressure, pulse, temperature, age, weight, height, sex, duration of the surgery and transfusion volume have been used to define altogether 25 attributes. Altogether 82 children patients were thoroughly examined in the study. Distribution between the two decision classes were: 88% with the tendency towards acidosis and 12% without the tendency towards acidosis. A summary of the MRACID database is presented in table 1.

5.3 AV database

Continuous time signals, like EEG, are often used in medicine by the medical staff to provide some information about the state of the patients. The objective of this study was to recognize patients with Alzheimer disease based on the recorder EEG signals. EEG or electroencephalogram is a record of electrical activity in the human brain. The EEG is one of the tools a neurologist uses in the diagnosis of brain conditions. There are several methods to analyze continuous time signals in medicine. Data mining techniques and data-based classification methods are not appropriate to deal with these kind of data. However, applying some domain knowledge enable one to extract parameters from EEG signals, that provide enough distinguishable information about a specific signal (Figure 2). In this way a set of continuous attributes are obtained, based on the domain knowledge, which can be used to construct a decision tree (avg, min, max, median in the frequency domain, avg, median, max, balanced median in the time domain, etc.). Distribution between the two decision classes was very unbalanced: 88% of patients with Alzheimer disease and 12% with some other vascular problems.

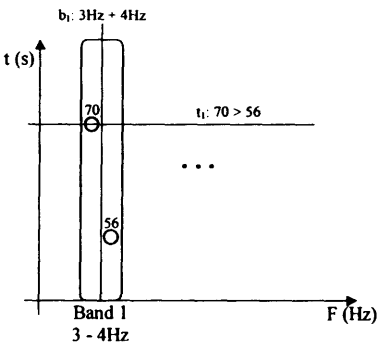


Figure 2: Selection of attributes from an EEG signal (band 1), showing the characteristic points for the frequency (b1) and time (t1) domain

5.4. Lipids database

The aim of the study was to determine blood cholesterol level of children without any invasive medical examination that can cause anxiety of the children. The database concentrates on the obese (by Roher index) children to determine the blood cholesterol level with simple non-invasive medical examination. The database consists out of 147 instance with 14 attributes (gender, weight, height, head circumference, chest circumference, etc.) and two very unbalanced possible decision: normal level (85%) and abnormal level (15%) blood cholesterol.

Other databases have been downloaded from the on-line repository of machine learning datasets maintained at the University of California at Irvine. We compared our multimethod approach (MultiVeDec) with four conventional approaches – C4.5 [8], C5/See5, Boosted C5 and genetic algorithm without multimethod extensions. The results are presented in table 1.

	C4.5		C5		C5 Boost		Genetic		MultiVeDec	
	Δ	□	Δ	□	Δ	□	Δ	□	Δ	□
av	76,7	44,5	81,4	48,6	83,7	50,0	90,7	71,4	93,0	78,7
breastcancer	96,4	94,4	95,3	95,0	96,6	96,7	96,6	95,5	96,6	95,1
heartdisease	74,3	74,4	79,2	80,7	82,2	81,0	78,2	77,8	78,2	78,3
Hepatitis	80,8	58,7	82,7	59,8	82,7	59,8	86,5	62,1	88,5	75,2
mracid	94,1	75,0	100,0	100,0	100,0	100,0	94,1	75,0	100,0	100,0
Lipids	60,0	58,6	66,2	67,6	71,2	64,9	76,3	67,6	75,0	73,3
Mvp	91,5	65,5	91,5	65,5	92,3	70,0	91,5	65,5	93,8	84,3

Δ accuracy □ average class accuracy

Table 1: A comparison of the multimethod approach to conventional approaches

The success of multimethod approach can be explained with the fact that some methods converge to local optima. With combination of multiple methodologies better local (hopefully global) solution can be found. For example when using decision trees the genetic methodology determines the attributes near the top and on the nodes near leafs the greedy method with different impurity measures can be used to reduce search space.

6. Conclusion

In this paper we introduced a multimethod approach to combine classifiers. The existing hybrid systems usually work sequentially or in parallel in the fixed structure and order performing whole tasks. On the contrary multimethod approach works simultaneously with several methods on a single task, i. e. some part are induced with different classical heuristics, some parts by hybrid methods and other part by evolutionary programming.

Presented multimethod approach enables quick and modular way to integrate methods into existing system and enables the simultaneous application of several methods. It also enables partial application of method operations to improve and recombine methodologies and has no limitation to the order and number of applied methodologies. It encourages standardization of knowledge representation and unifies the operational view on the methods. We also introduced a population of classifiers and meta level of multimethod approach to find appropriate order of method application with different parameters for a specific problem, that relieves the user from guessing the parameters.

References

- [1] S. Thrun and L. Pratt, editors. *Learning to Learn*. Kluwer Academic Publishers, 1998
- [2] T. G. Dietterich. *Ensemble Methods in Machine Learning*. In J. Kittler and F. Roli (Ed.) *First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science* (pp. 1-15) New York: Springer Verlag, 2000
- [3] K. McGarry, S. Wermter, J. MacIntyre The Extraction and Comparison of Knowledge From Local Function Networks *International Journal of Computational Intelligence and Applications*, Vol. 1 Issue 4, pp: 369-382, 2001
- [4] C. J. Iglesias. The Role of Hybrid Systems in Intelligent Data Management: The Case of Fuzzy/neural Hybrids, *Control Engineering Practice*, Vol. 4, no. 6, pp 839-845, 1996.
- [5] L. Todorovski, S. Dzeroski. Combining Multiple Models with Meta Decision Trees. In *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2000; 54-64
- [6] C. J. Merz. Using Correspondance Analysis to Combine Classifiers. *Machine Learning* 36(1/2): 33-58 Kluwer Academic Publishers, 1998
- [7] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, Reading MA, 1989.
- [8] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann publishers, San Mateo, 1993.
- [9] M. Šprogar, Peter Kokol, Špela Hleb Babič, Vili Podgorelec, Milan Zorman, *Vector decision trees, Intelligent data analysis*, vol. 4, no. 3,4, pp. 305-321, 2000
- [10] V. Podgorelec, Peter Kokol, R. Yamamoto, G. Masuda, N. Sakamoto, Knowledge discovery with genetically induced decision trees, In *International ICSC congress on Computational intelligence: methods and applications (CIMA'2001)*, June 19-22, 2001, University of Wales, Bangor, Wales, United Kingdom, ICSC
- [11] V. Podgorelec, P. Kokol. Evolutionary decision forests - decision making with multiple evolutionary constructed decision trees, *Problems in applied mathematics and computational intelligence*, pp. 97-103, World Scientific and Engineering Society Press, 2001.
- [12] M. Zorman, P. Kokol, V. Podgorelec, Medical decision making supported by hybrid decision trees, *Proceedings of the ICSC symposia on Intelligent systems & applications (ISA'2000)*, December 11-15, 2000
- [13] D. Whitley. An overview of Evolutionary Algorithms: practical issues an common pitfalls. *Information and software technology* vol. 43, pp. 817-831, 2000

Feature Extraction by Distance Neural Network in Classification Tasks

Cristian Maturana & Richard Weber

Department of Industrial Engineering, University of Chile, República 701

cristian@maturana.net

rweber@dii.uchile.cl

Abstract: Feature extraction is an important task for data mining applications, in particular for classification. On one hand it leads to a better understanding of the relations between features and classification results. On the other hand it helps to perform fast classifications with a reduced number of features. Principal component analysis (PCA) is one of the mostly used techniques for linear feature extraction. Recently neural networks have been proposed for non-linear feature extraction outperforming PCA in many cases like non-linear principal component analysis (NLPCA). The mentioned approaches have in common an error reduction when reproducing the initial feature space from the reduced space. We present a new approach which tries instead conserving the patterns distribution from the original space to the reduced space. This model called dNN (Distance Neural Network) provides very good results for several cases outperforming PCA and shows to be competitive with the best non-linear techniques for feature extraction.

1. Introduction

Feature extraction plays an important role in many data mining applications. Linear techniques like e.g. Principal Component Analysis (PCA) provide good results in general but suffer from the fact that they are based on linear models. Neural networks as non-linear techniques for feature extraction offer improvements compared to PCA in some cases. These techniques have in common a reduction of the error when reproducing the original attribute space after having determined the most relevant features as a combination of the original attributes. This is, however, only one aspect of feature extraction. Another very important aspect, which so far has not been considered in literature, is the conservation of the pattern distribution in the reduced space. We present dNN, the Distance Neural Network, which implements explicitly the objective of structure conservation. Using the implementation of our model we could show advantages of this new focus of feature extraction compared to the traditional techniques. The focus proposed in our model allows understanding the pattern distribution and the way the features are formed from the original attributes.

Section 2 describes formally the problem of feature extraction. In section 3 we present dNN the new model for feature extraction conserving the pattern distribution in the reduced space. Experiments and numerical results are shown in section 4. Section 5 provides a discussion of the presented work and conclusions.

2. Feature Extraction

We define formally the feature extraction problem, the notation used in literature and in the present work. We further describe non-linear principal component analysis (NLPCA) by neural networks as a feature extraction model.

2.1. Feature Extraction formulation

Given a set of P patterns $\{X_j\}_{j=1..P}$ distributed according to an unknown density function, with $X_j \in \mathcal{R}^n, \forall j=1..P$ and n the input space dimension. The n attributes that define the patterns in the input space, are identified by the index $i=1..n$. The goal is to find functions ψ and Φ_ψ from eq. (2.1) (according to the notation in literature).

$$X_j = \Psi\{\Phi_\psi(X_j)\} + \varepsilon_j \quad j=1, \dots, P \quad (2.1)$$

The function $\Phi_\psi: \mathcal{R}^n \rightarrow \mathcal{R}^m, 1 \leq m < n$, represents a reduced output space (\mathcal{R}^m) into which the patterns ($X_j \in \mathcal{R}^n$) are projected. According to [14] and [15] this means that Φ_ψ is a function that reduces the dimension of the input space through feature extraction, defined by a new group of m attributes or extracted features as a combination of the n original ones. The calibration of this function depends on the group of patterns $\{X_j\}_{j=1..P}$ and on the function ψ , which allows the reconstruction of the patterns starting from the reduced space to the original space. This dependence is important, since the calibration tries to obtain approaches that generate the smallest possible error, through the composition of both functions.

On the other hand, the function $\psi: \mathcal{R}^m \rightarrow \mathcal{R}^n$, is a continuous function, that locates the patterns from the reduced space \mathcal{R}^m , to the original space \mathcal{R}^n . This function reconstructs the input space of attributes and makes the estimation of the X_j vector.

Eq. (2.2) is considered as an estimation of X_j , where the error between X_j and \hat{X}_j (estimation of X_j), should be minimized, i.e. we want to minimize the expression in eq. (2.3).

$$\hat{X}_j = \Psi\{\Phi_\psi(X_j)\} \quad (2.2)$$

$$\varepsilon_j = \|X_j - \hat{X}_j\| \quad (2.3)$$

More formally, as outlined e.g. in [1] we try to determine the functions ψ and Φ_ψ , such that the expression in eq. (2.4) is minimized.

$$\text{Min}_{\Psi, \Phi_\psi} \sum_{j=1}^P \|X_j - \Psi\{\Phi_\psi(X_j)\}\|^2 \quad (2.4)$$

Solving the optimization problem given in (2.4) provides a function ψ that estimates the vector X_j , according to the least square criteria. The function Φ_ψ reduces the dimension of the original feature space through feature extraction. Both functions together allow to find the best estimator that reconstructs the patterns in the input space, in the sense of the least square criteria.

2.2. Non Linear Principal Component Analysis

The non-linear principal component analysis NLPCA, is an auto-associative neural network model that reduces the dimension of the original space through feature extraction in a non-linear form. Its architecture is a feedforward auto-associative neural network of five layers

as shown in figure 2.1 [14]. The third layer (known as bottleneck layer in literature) has less units than the input and output layers. The input, bottleneck and output layer have linear activation functions. The second and the fourth layers have a sigmoid activation function, and are known as encoder and decoder layers, respectively. These two layers cause the non-linear characteristic. If we denote L as the linear activation function, S as the sigmoid activation function (nonlinear), n as the number of units in the input and output layer, $m < n$ as the number of units in the bottleneck layer, u and v as the number of units in the encoder and decoder layers, respectively, the NLPCA architecture is given by $nL uS mLvSnL$ (where each layer is defined by the number of its units and next its activation function). This network can end up being more and more complex and non-linear, adding more hidden layers with non-linear activation functions; see e.g. [17].

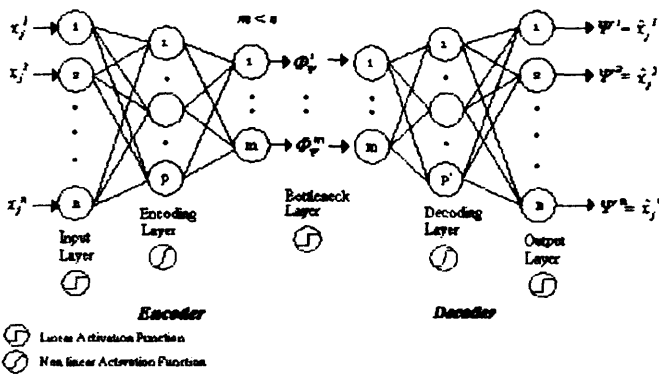


Figure 2.1: Non Linear Principal Component Analysis Neural Network NLPCA

The hidden layer compresses the information from the original space (dimension n) into a smaller dimension (m). This model is able to learn non-linear representations, NLPCA obtains similar results as principal component analysis (PCA) [4]. Following [14], NLPCA is a direct generalization of PCA through feedforward neural networks.

The NLPCA network trains with an error function given by eq. (2.5) through the back-propagation algorithm learning rule (the desired output coincides with the input pattern). After training of the network, the weight adjustment allows to determine the pattern position in the output space through the input, encoder and bottleneck layer.

$$E_j = \|X_j - \hat{X}_j\|^2 \quad (2.5)$$

NLPCA can be divided into two virtual networks (trained as only one): Encoder network and Decoder network. Input, encoder and bottleneck layer compose the first one, whereas the Decoder network contains the bottleneck, decoder and output layer. NLPCA estimates the non-linear functions ψ and Φ_ψ from eq. (2.4), by Encoder and Decoder network, respectively (fig. 2.1), as a composition of functions between the input, encoder and the bottleneck layer for ψ on one hand, and between the bottleneck, decoder and output layer for Φ_ψ on the other hand.

The NLPCA network has several advantages like the possibility to find non-linear relationships among the input space patterns and to find spaces with smaller dimension by feature extraction.

3. Distance Neural Network Approach for Feature Extraction

A new neural network model is developed for Feature Extraction. This network, called dNN: Distance Neural Network, is based on the conservation of distances between patterns in original space and the reduced space defined by feature extraction.

The feature extraction problem, as it has been described before, has been treated by auto-associative neural networks, and specifically by NLPCA. The training of such neural networks is based on the reconstruction of the patterns in the original space compressing their information in the hidden layer (with $m < n$ units). The extracted features are a result of the network's capacity to reconstruct the patterns in the original space preserving the most information possible.

dNN is a neural network with an architecture similar to NLPCA that trains in two phases: the first phase or Encoder (with an architecture $nLmS$), to obtain the new features, and the second one or Decoder (with an architecture $mSuS:nL$), to reconstruct the input space (fig. 3.1). The objective of the Encoder is to extract the new features through minimizing an error function based on the difference among the distances between patterns in the different spaces (original and reduced space). The training of the Encoder is through a learning without supervision, because the desired output (features to be extracted) is not known beforehand. In other words, the new features are calculated in the reduced space through the weights of the net, and the result (patterns position in the reduced space) is used to calculate the distances between patterns. In turn, this distance is used for the calculation of the error function and the backpropagation algorithm. Following the NLPCA description, the dNN architecture is given by $nLmSuSnL$ (fig. 3.1).

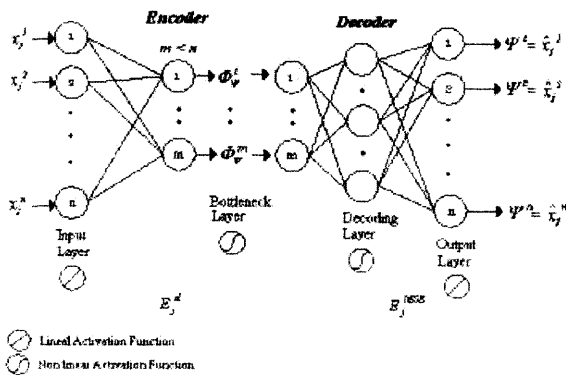


Figure 3.1: Autoassociative Distance Neural Network dNN

The objective of the Encoder is to maintain the pattern distribution in the original space, pursuing each pattern j projected into the reduced space, i.e. it maintains the distances among the object j and all the other objects $k \neq j$ in the reduced space.

To reach this target, eq. (3.1) shows the error function proposed for dNN. The intention of eq. (3.1), is to maintain the relationship of the distances among the patterns in the original space and in the reduced space defined by the extracted features.

$$E_j^d = \frac{\sum_{k \neq j}^P (d_{jk} - D_{jk})^2}{2 \sum_{j=1}^P \sum_{k > j}^P D_{jk}} \quad (3.1)$$

Where P is the number of patterns, D_{jk} the square distance in the input space, among the pattern j and k defined in eq. (3.2), d_{jk} the square distance in the output space defined by the extracted feature, among the pattern j and k defined in eq. (3.3).

Square distance between patterns j and k in the input space:

$$D_{jk} = \frac{1}{n} \|X_j - X_k\|^2 \quad (3.2)$$

Square distance between patterns j and k in the output space:

$$d_{jk} = \frac{1}{m} \|\Phi(X_j) - \Phi(X_k)\|^2 \quad (3.3)$$

In (3.2), n corresponds to the dimension of the original space and in (3.3) m is the dimension of the reduced space. Error function (3.1) conserves the structure of the input space through the minimization of the difference among the distances between patterns of the different spaces (original and reduced space). The error function defined in (3.1) and some variations have been used in [4], [11] and [13], for other types of applications like multidimensional scaling (MDS) and for Sammon's Mapping. In [11] an implementation of this error function E_j^d for non-linear principal component analysis is presented. The present work differs from [11] in that the defined error function in (3.1) will be used only to train the Encoder network.

On the other hand, the objective of the Decoder (second phase of dNN), is to reconstruct the input space starting from the new extracted features from the Encoder. The Decoder trains in a second phase with the error function defined in eq. (3.4), once the Encoder is already trained. Consequently, dNN training is a sequential training with different error functions.

$$E_j = \frac{1}{2} \|\Psi(\Phi_\Psi(X_j)) - X_j\|^2 \quad (3.4)$$

The net dNN, will be trained through the backpropagation algorithm in two sequential phases. Results of the Encoder, derived by the error function (3.1), will be used to train the Decoder with the error function defined in eq. (3.4) in order to reconstruct the original space from the extracted features defining the reduced space.

The intention of the dNN separation in an Encoder network and in a Decoder network is purely methodological. In NLPCA it is necessary to make the difference between the Encoder and Decoder because the entire net extracts the features. Here, only the Encoder is the

feature extractor, and we use the Decoder to compare the results and test the features' quality to reconstruct the original patterns.

4. Experiments and Results

We will test the performance of the proposed model (dNN) and will compare the results between dNN, NLPCA, and PCA. For this purpose we first define a performance measure. Then we describe the data sets we used and the respective results.

4.1. Performance Measures

In [9], [15] and [14] the NMSE performance measure (Normalized Mean Square Error) is proposed. The NMSE indicates the error of a model when estimating the patterns. The NMSE range is $[0, \infty)$, being a perfect estimation with $\text{NMSE}=0$, estimating the mean gives $\text{NMSE}=1$ and worse estimations have $\text{NMSE}>1$.

$$\text{NMSE} = \frac{\sum_{j=1}^P \|X_j - \hat{X}_j\|^2}{\sum_{j=1}^P \|X_j - \bar{X}_j\|^2} \quad (4.1)$$

Eq. (4.1) shows the NMSE: $X_j \in \mathcal{R}^n$ represents the input pattern, $\hat{X}_j \in \mathcal{R}^n$ represents the model estimation of X_j , and $\bar{X}_j \in \mathcal{R}^n$ represents the patterns mean in the input space. The classification task will be considered as a performance measure according to [4] and [1]. We use a multilayer neural network (MLP) for the classification task based on four different feature sets: the original features, features extracted by PCA, NLPCA and by dNN.

4.2. Data Set and experiment

The data set we used describes clients of a Chilean bank. Twenty descriptive attributes characterize each client (i.e., $n=20$), and they belong to one of 2 classes: clients with an open current account, and clients that closed their account. The data set contains 1385 patterns, with 700 patterns (50.54%) belonging to the first class (clients with an open account), and 685 patterns (49.46%) belonging to the class of clients with closed accounts. The objective is to identify clients that are about to close their accounts. The data set stems from a research project at the Department of Industrial Engineering, and is described in the appendix.

The experiment's objective is to classify the clients using a multilayer perceptron neural network, with all attributes, with $m=10$ extracted features obtained by PCA, with $m=10$ extracted features obtained by NLPCA, and with $m=10$ extracted features obtained by dNN.

4.3. Results

For the classification task we use a multilayer perceptron (MLP) with the following architecture: 10S:7S:2S. For the feature extraction models, table 4.1 shows the parameters we used for training of NLPCA and dNN, respectively.

Table 4.1. Feature Extraction Models: Parameters for Training

$n : m = 20 : 10$	NLPCA	dNN (Encoder)	dNN (Decoder)
Architecture	20L:15S:10L:15S:20L	20L : 10S	10S : 15S : 20L
Training Epoch	8000	150	1000

Concerning NMSE both non-linear models obtain excellent results: NLPCA obtains 0.0239 and dNN obtains 0.0588. PCA gives a NMSE of 0.4642, out of the chart range shown on the left side of Figure 4.1

Classification by multilayer perceptron (MLP) provides the results shown on the right side of Figure 4.1. Using all the original attributes, the MLP gives 76.96% of correct classification, while the same technique using the 10 features extracted by PCA gives 72.48%. For NLPCA we achieved 82.63% and for dNN 81.13% in correct classification using in both cases $m=10$ extracted features.

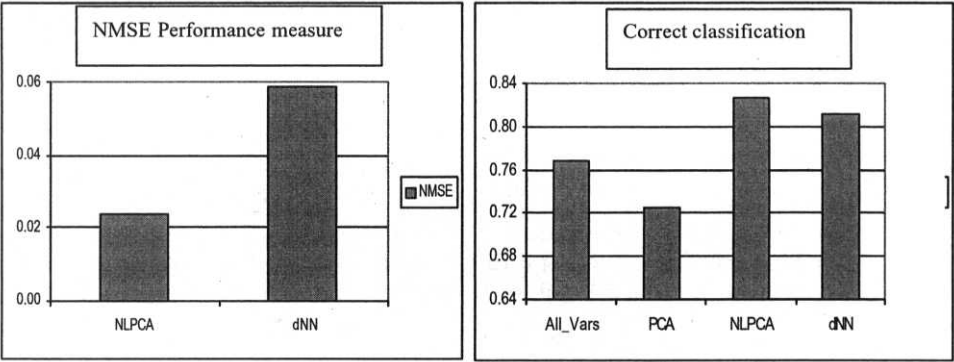


Figure 4.1. NMSE and Correct classification (by MLP)

To analyze how the extracted features are composed starting from the original attributes, we consider the contribution of each attribute to each feature, i.e., the importance of the original attributes for the extracted features. Taking the set of original attributes and its contribution to each feature, we determined the standard deviation of the contribution for each feature. Figure 4.2 shows these standard deviations of attribute contribution for each extracted feature (FE1 , ..., FE10), for every model (PCA, NLPCA, dNN).

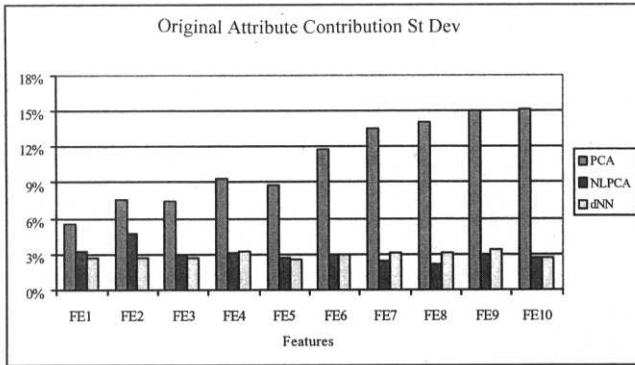


Figure 4.2. Standard Deviation of original attribute contribution to each Feature extracted

A high standard deviation means that the feature is composed of few attributes. On the other hand, a small standard deviation means that the respective feature is composed equally of all the original attributes. In the case of PCA, the average standard deviation is 10.82% (as shown in figure 4.2), because this model finds orthogonal components into which the features are projected. On the other hand, the average of the standard deviation for the NLPCA model is 3.01%, because this model does not extract orthogonal components, but for each feature, the model extracts more information from the original attributes. For the dNN model, the average is 2.94%, i.e. a better perform than PCA and NLPCA, because of the model objective, as declared in (3.1). This result demonstrates that dNN is a model that extracts the most important information from the original space taking all attributes into consideration. With our model dNN we have presented a technique for feature extraction that tries to conserve explicitly pattern distribution from the original space. The presented results indicate that using all the attributes homogeneously gives better classification results than using few attributes for feature construction, as for example in PCA.

5. Discussion and Conclusion

From the experiments it can be concluded that dNN is a highly competitive model for feature extraction. While NLPCA gives better results regarding the NMSE measure, dNN outperforms PCA in classification tasks. Having in mind that dNN uses an objective function that does not explicitly minimize the error in the original space (measured e.g. by NMSE) it even obtains excellent results regarding NMSE.

Concerning correct classification (applying MLP), the model dNN outperforms PCA and gives results slightly worse than NLPCA. This confirms the competitiveness of the model for the feature extraction problem. The results demonstrate that dNN is a valid model for feature extraction, and also that the error function in (3.1) allows a good representation of the original patterns in a reduced space through the extracted features, reflected in the classification task. On the other hand, dNN is a simple model to check the way the features are extracted from the original attributes.

The present work opens directions for future developments based on dNN. Various error functions that incorporate the conservation of the pattern distribution in the original and reduced space can be tested. Furthermore, different distance functions could be analyzed and new learning algorithms for feature extraction could be developed based on the presented work.

6. References

- [1] Albanis G., Batchelor R., "Predicting High Performance Stock Using Dimensionality Reduction Techniques based on Neural Networks", *Technical Report, City University Business, 2000*
- [2] Berthold M., Hand D., "Intelligent Data Analysis: an introduction", *Cap. 7, Springer-Verlag Ed., 1999*
- [3] Carreira M., "A Review of Dimension Reduction", *Technical Report CS 9690, Dept of Computer Science, University of Sheffield, 1997*
- [4] De Backer S., Naud A., Schunders P., "Non-Linear Dimensionality Reduction Techniques for Unsupervised Feature Extraction", *Pattern Recognition Letters, Vol.19, pp. 711-720, 1998*
- [5] De Mers D., Cottrel G., "Non-Linear Dimensionality Reduction", *Advances in Neural Information Processing System, vol. 5, pp. 580-587, 1994*
- [6] Devulapalli S., "Non-Linear Principal Component Analysis Classification of EEG during Mental Task", *Thesis, Degree of Master of Science, 1996*
- [7] Edelman Sh., Intrator N., "Learning as extraction of low-dimensional representations", *Perceptual Learning R. Goldstom and D. Medin an P. Shyns(Eds) 36, pp. 353-380, 1997, Academic Press*
- [8] Estévez P., "Clasificación de Patrones mediante Redes Neuronales Artificiales", *Anales del Instituto de Ingenieros de Chile, pp. 24-31, Abril 1999*
- [9] Estévez P., "Selección de Características para Clasificadores Neuronales", *Anales del Instituto de Ingenieros de Chile, pp. 65-74, Diciembre 2000*
- [10] Fisher R., "The use of multiple measurements in taxonomic problems", *Annual Eugenics, 7, Part II, 179-188, 1936*
- [11] Garrido L., Gómez S., Roca J., "Improved Multidimensional Scaling Analysis Using Neural Networks with Distance-Error Backpropagation", *Neural Computation, No. 11, pp. 595-600, 1999*
- [12] Kambhatla N., Leen T., "Fast-non-linear Dimension Reduction", *Advances in Neural Information Processing Systems, No. 6, pp.152-159, 1995*
- [13] Lerner B., Guterman H., Aladjem M., Dinstein I., "A Comparative Study of Neural Network Based Feature Extraction Paradigms", *Pattern Recognition Letters, vol. 20(1), pp. 7-14, 1999*
- [14] Malthouse E., "Some Theoretical Results on Nonlinear Principal Component Analysis", *Technical Report, Northwestern University, 1998*
- [15] Monahan A., "Non-Linear Principal Component Analysis by Neural Networks. Theory and Application to the Lorentz System", *Journal Climate, Vol.13, pp. 821-835, 2000*
- [16] Sengupta S., Boyle J., "Non-Linear Principal Component Analysis of Climate Data", *PCMDI, Program for Climate Model Diagnosis and Intercomparision, Report No. 29, 21 pp., 1996*
- [17] Shajith M., Misra H., Yegnanarayana B., "Analysis of Autoassociative Mapping Neural Networks", *Int. Joint Conf. On Neural Networks, July 2000*
- [18] Zell A., et al., "SNNS Stuttgart Neural Network Simulator", *User Manual Version 4.2, University of Tübingen, Department of Computer Architecture, 1998*

Appendix

Statistical description of the data set used in the experiments, corresponding to bank customers. For competitive reasons attributes are given as A1, ..., A20.

Attribute	Min	Max	Mean	Std. Deviation
A1	21	86	41.3466	12.0395
A2	0	1	0.6700	0.4704
A3	-1	1	0.1567	0.8140
A4	5.5215	8.5942	6.4318	0.5109
A5	0	1	0.3278	0.4696
A6	3	366	55.8621	52.5406
A7	0	417	38.5906	36.2388
A8	0	390	37.2274	34.0053
A9	0	391	42.9271	37.4646
A10	-13.2507	17.9230	10.6907	5.5482
A11	-13.2303	18.0704	10.6844	5.7030
A12	-13.5381	17.3818	10.8536	5.6142
A13	-14.1224	18.4021	11.2472	6.0686
A14	-15.1544	18.8852	11.5351	6.8580
A15	-13.7737	19.0512	8.1519	7.6611
A16	-37.57	66.61	1.0962	3.9946
A17	-146	488.38	6.0975	25.4247
A18	-242.84	649.58	3.5305	22.8696
A19	-113.49	487.27	2.7944	20.5819
A20	-357.55	585.73	2.4719	26.4875

Revealing Feature Interactions in Classification Tasks

Derek Partridge & Shuang Cang
Department of Computer Science
University of Exeter
Exeter EX4 4PT, UK
D.Partridge/S.Cang@exeter.ac.uk

Abstract. This paper presents a contribution to the theory of optimal feature-subset selection associated with pattern recognition or classification tasks. It extends the theory of Mutual Information (MI) to deal with the difficulties introduced by feature interaction. The essential contribution is to permit MI calculations between sets of features and the target class such that all interactions between the features in the chosen set are taken into account in the MI value produced. In order to accomplish this extension from traditionally pairwise (i.e., feature-feature or feature-class) MI computation we have developed algorithms to transform any continuous-valued feature into a discrete-valued one, and to transform any set of discrete-valued features into a 'composite' feature suitable for the necessary MI calculations. We have built these algorithms into classical forwards and backwards sequential search procedures, and these provide an initial survey of the interactions present among features within a given set of feature vectors. If no significant feature interaction is present then the features have effectively been ranked, and the optimal subset selection problem has been solved. When feature interactions are present, the initial survey will indicate where and what interactions are present and will suggest, if necessary, further probes with the extended MI algorithm to reveal their full nature.

We demonstrate the effectiveness of the extended MI algorithm on a number of examples that have been presented as problematic in the literature, especially in the feature-selection literature that has employed MI theory .

1. Introduction

Classification tasks are typically managed in terms of sets of feature values extracted from examples of the problem, and these are to be used to discriminate between a set of potential classifications for any new set of features. Issues of efficiency, accuracy and even possibility for the desired classifier can be heavily reliant on effective pre-processing of the initial set of potential features. Typically, we need to reduce the initial features set to an optimal subset by selecting the most important features, or by eliminating the least important features. This is the problem of feature selection. The reason why this can be a difficult problem is that features may interact so that feature importance (i.e., discriminatory power with respect to the target classes) is not an ordinal property, and noise within the feature values may mask or even overwhelm the discriminatory value of certain features.

So much for the reasons why the feature-selection problem is a difficult one, the next question is: why do we need to solve it? There is no one simple answer. It may be necessary to reduce a very large number of initial features to a more manageable size for purely practical computational reasons: it is not only quicker and cheaper to build and operate a classifier with less features, it is also cheaper and easier to collect less feature values. It may be that the initial set is a best guess as to what features will prove to be important. Such a feature set is likely to contain irrelevant or redundant

features in which case efficiency and cost concerns again suggest that the non-important features be found so that the usable feature set is minimised. Finally, noise in the features values, which may introduced through measurement or transcription techniques, can mean that certain features have an adverse effect on classifier accuracy. Underlying all feature set reduction efforts is the basic assumption that a minimal feature set with maximum discriminatory power will also be optimal for classifier accuracy.

The variety of approaches to the problem of feature subset selection have been variously classified. Theodoridis and Koutroumbas [1] present a two-way split into methods that treat features individually and those that do not, i.e., those that focus on techniques measuring classification capabilities of feature vectors. As they point out: treating features individually has the advantage of computational simplicity but may not be effective for complex problems and for features with high mutual correlation. If we add the possibility of noise (as described above) then we have a characterization of the sort of pattern recognition task that we wish to focus on. Thus it is the latter category of feature subset selection methods that are of interest.

Within this category, Theodoridis and Koutroumbas focus on suboptimal techniques as the alternative of assessing all possible vector combinations is impractical for all but the most trivial problems. Two well established suboptimal techniques are Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS) both of which select features, and accept or reject them one at a time in an attempt to minimize the number of feature vectors evaluated whilst accommodating some aspects of mutual correlation between features. But both techniques suffer from the so-called 'nesting effect' --- once a feature has been accepted or rejected it cannot be reconsidered in the context of a different feature vector.

An improvement, which increases computational cost, uses a 'floating search' (Pudil, Novovicova, and Kittler [2]) --- this encompasses the flexibility to reconsider previously accepted or rejected features. It can be added to either SFS to give Sequential Floating Forward Selection (SFFS) or to SBS to give Sequential Floating Backward Selection (SFBS).

Jain and Dongker [3] present a taxonomy of feature-selection algorithms which bifurcates at the root to cluster "statistical pattern recognition" (SPR) techniques on one branch and those using "artificial neural networks" (ANN) on the other. They further divide the SPR techniques into "optimal" and "suboptimal", and this latter branch encompasses SFFS and SFBS under the heading "deterministic single-solution suboptimal" techniques.

Within these frameworks, the technique to be described in this paper is based on an SPR technique that is suboptimal, although current indications are that its results are often optimal, in practice. Previously (Cang and Partridge [4]), we have defined and tested two complementary algorithms, one based on SFS and one on SBS. The key decision, i.e., on what basis is feature importance judged, is determined by the quantity known as Mutual Information (MI). MI is the amount of information that one random variable contains about another random variable. MI has previously been used as the basis for feature selection algorithms (Battiti [5]; Kwak and Choi [6]) but only in its straightforward form to assess the MI between pairs of variables --- between features or between a feature and the class. This pairwise use of MI would, of course, fall foul of the above-mentioned problems with the sequential approaches, which the 'floating' enhancement lessens but does not solve. These are the problems of feature interaction which can mean that any pairwise-based assessment may produce badly suboptimal results. This potential fault is recognized by the previous

users of MI and they introduce a new parameter, β , to attempt to compensate for redundancy between features. However, a single value, $\hat{\alpha}$, can only be, at best, a rough approximation to the variety of feature-feature interaction possible in any given classification task, especially when no systematic procedure for choosing a good value for $\hat{\alpha}$ is suggested.

The basic innovation that we introduced was to enable the computation of MI between a *set of features* and the class. In this approach any feature interaction that exists within the set under consideration is taken into account in the MI result produced. By first surveying the full set of potential features in both forwards and backwards sequential modes, we obtain the information necessary to determine what further (if any) MI calculations to make in order to reveal the feature interactions of interest.

2. The Preliminary Survey

The two basic survey algorithms employed are called Mutual Information Forward Search Feature Selection (MIFSFS) and Mutual Information Backward Search Feature Selection (MIBSFS). One further hitch with the basic equations of MI is that in general the exact computation for continuous-valued features¹ cannot be performed. So in order to make our two survey algorithms applicable to all classification tasks, we also developed algorithms for converting any continuous-valued feature into a discrete-valued feature.

The key innovations that make the forwards and backwards surveys valuable (using MIFSFS and MIBSFS) are:

1. Preprocessing of all continuous-valued features to convert them into discrete-valued features.
2. A procedure to convert any set of discrete-valued features into a single composite feature.

The direct consequence of these two conversion procedures is that the standard equation for MI can be used to determine the amount of information that any feature set contains about the class or about other features, or feature sets. By adjustment of both the feature sets used and the comparisons made, it is thus possible to unravel the feature interactions that exist in any classification problem.

This preprocessing of continuous-valued features involves a procedure for determining the optimal number of components for a Gaussian mixture model (Cang and Partridge [7]). An error function, $E(bin)$, for the resultant probability distribution, $f(x)$, can be computed with respect to the number of discrete bins, bin , used to approximate it --- i.e., a histogram representation (Cang and Partridge [7]). When bin is small, $E(bin)$ is large because a minimal histogram will be a poor approximation to the derived distribution $f(x)$. As the number of bins increases, $E(bin)$ decreases. There is a lower bound on the number of bins called $Lbin$. When $bin > Lbin$, $E(bin)$ does not decrease significantly. $Lbin$ is the optimal cut off point of the error function. Thus we use the $Lbin$ histogram to provide a discrete approximation of the probability density function $f(x)$. This simple approach to a difficult problem is not perfect, but it seems to be good enough as the results demonstrate.

Construction of a single composite feature from a set of n discrete-valued features ($F = \{f_i\}$ where $1 \leq i \leq n$) is a process of finding all combinations of discrete values selected one from each of the individual features, f_i . If feature f_i can take a

¹ Although all representations are discrete within a conventional computer, many such as the real numbers are so patchy and hardware specific that they are best treated as in effect continuous.

maximum of k_i different discrete values, the set of f_i will produce a composite feature with a maximum of $\prod_{i=1}^n k_i$ different discrete values. However, as the composite feature will only contain those combinations of values that actually occur in the component f_i , the theoretical maximum is seldom attained.

The algorithms MIFSFS and MIBSFS then step through the initial feature set in opposite directions adding or deleting (respectively) one feature at a time, and the basis for the selection or de-selection of a feature is the change in MI that results. The key point being that this change in MI is computed in the context of a set of features within which all interactions are operative while the importance of the next feature to be added or deleted is assessed. The basic equations are given in the Appendix while formal treatments can be found in Cang and Partridge [4].

The algorithms each produce a subset of the most important features (i.e., of those subsets chosen, it is the one with the highest MI in relation to the class), and a ranking of the individual features with respect to their importance. However, the MI values that give this ranking are in terms of the change in MI caused by that feature in the context of the complete subset of higher-ranked features. When the two algorithms differ in either subset selection or feature ranking within the subsets (or both), this is indicative of feature interactions, and it provides a focus from which to begin to reveal the details of these interactions.

3. The Proposed Algorithms

Two new algorithms, mutual information feature space forward selection (**MIFSFS**) and mutual information feature space backward selection (**MIBSFS**), will be presented in this section, and we can obtain the optimal feature space or best feature space.

3.1 MIFSFS algorithm for feature ranking

Step 1: For each continuous-valued feature to be considered, determine the number of components in the Gaussian mixture model by using algorithms in [7].

Determine the optimal cut-off point for the number of bins by using the $E(bin)$ function.

Step 2: Set selected feature set $S = \{ \}$; $F = \{\text{all features}\}$.

Step 3: For each feature space $S \cup f_i$, where $f_i \in F$, calculate the mutual information between the feature space and the class $I(S \cup f_i, C)$ and find $f_j = \max_i I(S \cup f_i, C)$.

Step 4: $F \leftarrow F \setminus \{f_j\}$, $S \leftarrow S \cup \{f_j\}$ and repeat from **Step 3** until $F = \{ \}$. Ranking the features given by ascending order of transfer to S .

3.2 MIBSFS algorithm for feature ranking

Step 1: For each continuous-valued feature to be considered, determine the number of components in the Gaussian mixture by using algorithms in [7], and determine the optimal cut-off point for the number of bins by using Error function (7).

Step 2: Set selected feature set $S = \{ \}$; $F = \{\text{all features which are need to be analysed}\}$.

Step 3: Eliminate each feature f_i from the feature space F , where $f_i \in F$ and calculate the mutual information between the feature space and the class $I(F \setminus f_i, C)$ and find $f_j = \max_i I(F \setminus f_i, C)$.

Step 4: $F \leftarrow F \setminus \{f_j\}$, $S \leftarrow S \cup \{f_j\}$ and repeat from **Step3** until $F = \{ \}$. Ranking of feature given by descending order of transfer to S .

3.3 Optimal Feature Space

After ranking for all features, choose the feature space S for which $\max_i I(S, C)$ reaches the maximum mutual information with respect to the class. This is the optimal feature subset.

We can also identify the irrelevant and redundant feature through the algorithms **MIFSFS** and **MIBSFS**. A feature is irrelevant if it can make no contribution to the class discrimination, so there is small (with noise) or zero (no noise) mutual information between the feature and the class comparing with mutual information between other features and the class. If it is difficult to determine whether or not a certain feature is irrelevant due to the noise problem, we can use the mutual information between this feature and the rest of the features to identify it. A feature is redundant to the extent that its contribution to the class discrimination can be made from a combination of the other features. In the next section, we give the results for two examples that have been presented as problematic in previous developments of MI for feature selection, and illustrate both redundancy and irrelevancy.

4. Results

In order to present a clear comparison, we consider two previously published examples and include the results of the two pairwise MI algorithms, **MIFS** and **MIFS-U**, both described in [6].

Example 1: This example is given in Kwak and Choi [6] , and we re-examine it. Each of the random variables X and Y are uniformly distributed on $[-1, 1]$, and there are three input features X , Y and X^2Y . The output belongs to class Z where,

$$Z = \begin{cases} 0 & \text{if } XY < 0 \\ 1 & \text{if } XY \geq 0 \end{cases}$$

We randomly select 2 different data sets, **A** and **B**, and each contains 1000 patterns. The numbers of components in the Gaussian mixture models are determined to be around 4, 4 and 3 for the features X , Y and X^2Y , respectively, using the algorithm in Cang and Partridge [7]. Following Kwak and Choi, we use 10 bins for X , Y and X^2Y . We also use 10, 8 and 8 bins for features X , Y and X^2Y , respectively, as indicated by the procedure in [7]. Representative results from the two survey algorithms are presented in Tables 1 and 2, where each successive addition (**MIFSFS**) or deletion (**MIBSFS**) of a feature is illustrated in a new row (and called a Run). In addition, these two tables give the feature ranking in parenthesis next to the MI value that determines rank for each feature. For Data set **A** (Table 1) the MI between the full feature space and the class was 0.9950, while for data set **B** (Table 2) it was 0.9915 as per the single Run 3 value in the two tables. In these two tables 10 bins were used for each feature.

Table 1: Successive MI values from **MIFSFS** (cols. 2-4) and from **MIBSFS** (cols.5-7)

Data set A	X	Y	X^2Y	X	Y	X^2Y
Run 1	0.0071	0.0027	0.0122(1)	0.0269	0.8632	0.9872(3)
Run 2	0.8632(2)	0.0269		0.0027(1)	0.0071(2)	
Run 3		0.9950(3)				

Table 2: Successive MI values from MIFSFS (cols. 2-4) and from MIBSFS (cols. 5-7)

Data set B	X	Y	X^2Y	X	Y	X^2Y
Run 1	0.0083(1)	0.0031	0.0028	0.0257	0.5731	0.9876(3)
Run 2		0.9876(2)	0.5731	0.0031(1)	0.0083(2)	
Run 3			0.9915(3)			

The resultant feature rankings, together with the results of **MIFS** and **MIFS-U**, are given in Table 3 where * in a table indicates that the feature is irrelevant or redundant.

Table 3: Feature Ranking for Example 1 Data

Feature	X	Y	X^2Y
Data Set A			
10 bins are used for each feature			
MIFS : $\beta=1$	2	3	1
MIFS-U : $\beta=1$	3	2	1
MIFSFS	2	3	1
MIBSFS	1	2	3*
10, 8 and 8 bins are used for each feature, respectively			
MIFS : $\beta=1$	1	2	3
MIFS-U : $\beta=1$	1	3	2
MIFSFS	1	2	3*
MIBSFS	1	2	3*
Data Set B			
10 bins are used for each feature			
MIFS : $\beta=1$	1	2	3
MIFS-U : $\beta=1$	1	3	2
MIFSFS	1	2	3*
MIBSFS	1	2	3*

Because of interaction between the features there are a number of equally good feature rankings. The optimal feature space contains two features, either X^2Y and X , or Y and X . No algorithm selects Y as the most important, but **MIFS-U** does incorrectly select it second to pair with X^2Y (second row Table 3). The only incorrect rankings pair X^2Y and Y as the two most important features. From Table 3, we see that, apart from the incorrect performance of **MIFS-U** already mentioned, all the algorithms otherwise perform correctly. However, only **MIBSFS** and **MIFSFS** also indicate that the third feature is redundant. Curiously, this redundancy appears to be obscured on the single occasion that X^2Y is selected first (row 3 Table 1), and this is also the situation when the **MIFS-U** algorithm falters. The obvious bias in Table 3 with respect to choice of most important feature (it could be any of the three but X appears favoured except when 10 bins used for all features of data set A) may in part be accounted for by choice of bin numbers, and we shall address this issue in the paper's conclusions.

Because the class labelling for this problem might alternatively be defined as,

$$Z = \begin{cases} 0 & \text{if } X^2Y/X < 0 \\ 1 & \text{if } X^2Y/X \geq 0 \end{cases}$$

we cannot understand Kwak and Choi’s [6] view that the problem with their algorithms (Table IV in [6]) is the fact that they select feature X^2Y as the most important. From our perspective the problem arises because their algorithms then selects Y as the second most important feature. This seems to be the clear error as any of the three features could equally legitimately be selected as first choice, but this first choice then constrains the second, and X^2Y followed by Y is wrong. The authors continue to suggest that the crux of this problem is the use of mutual information, whereas we would claim that it is not the use of mutual information as such (as our successful algorithms clearly demonstrate), but the pairwise basis for assessment whatever the underlying interaction assessment theory. Feature interaction can always seriously distort the outcomes from decisions made on the basis of only pairwise assessments.

Example 2: Monk3 Data With 5% Noise

The **Monk3 Data** example is also studied in [6] where it is presented with varying degrees of noise (0, 5 and 10 percent), but no details of how to obtain or reproduce their example data sets. The data set used here is collected from UCI (<http://www.sgi.com/tech/mlc/db/>) and is stated to contain 5% noise (misclassifications). There are 6 discrete-valued input features, and two output classes, “true” and “false”. There are 122 patterns in the data set. The “true” concepts underlying Monk3 is given by the logical operator $(F_5 = 3 \cap F_4 = 1) \cup (F_5 \neq 4 \cap F_2 \neq 3)$, where \cap and \cup denote logical AND and OR, respectively. Thus features F_1 , F_3 and F_6 are totally irrelevant to the discriminatory task.

Tables 4 and 5 give representative results obtained from the two survey algorithms. The ranking results for **MIFS**, **MIFS-U**, **MIFSFS** and **MIBSFS** are given in Table 6, where # indicates that we can keep or delete the associated feature from the best feature space as it accounts for only a very small amount of the total MI. Its worth is difficult to distinguish from noise.

Table 4: Successive MI values from MIFSFS on Monk3 data.

	F_1	F_2	F_3	F_4	F_5	F_6
Run 1	0.0071	0.2937(1)	0.0008	0.0029	0.2559	0.0071
Run 2	0.3281		0.3348	0.3579	0.7465(2)	0.3045
Run 3	0.8125		0.8120	0.8678(3)		0.7800
Run 4	0.9998(4)		0.9055			0.8931
Run 5			0.9998			0.9998

Table 5: Successive MI values from MIBSFS on Monk3 data.

	F_1	F_2	F_3	F_4	F_5	F_6
Run 1	0.9178	0.8071	0.9998(6)	0.9342	0.7517	0.9998
Run 2	0.8931	0.6760		0.8603	0.6012	0.9998(5)
Run 3	0.8678(4)	0.5539		0.8125	0.4584	
Run 4		0.3242		0.7465(3)	0.3579	
Run 5		0.2559(1)			0.2937(2)	

Table 6: Feature Ranking for Monks3 Data

Feature	F_1	F_2	F_3	F_4	F_5	F_6
MIFS: $\beta=1$	5	1	3	6	2	4
MIFS-U: $\beta=1$	4	1	5	3	2	6
MIFSFS	4*	1	5*	3	2	6*
MIBSFS	4#	1	6*	3	2	5*

From Table 6, we can see that algorithms **MIFS-U**, **MIFSFS** and **MIBSFS** give the correct feature ranking, but algorithm **MIFS** incorrectly selected F_4 as the least important. The results for **MIFS** and **MIFS-U** resemble the ones given as 10% noise in the earlier publication, and it may be that 10% noise generated by their means is closest to 5% noise generated for the UCI database. **MIFSFS** and **MIBSFS** correctly indicate that F_3 and F_6 are irrelevant or redundant while the judgement on F_1 is equivocal.

On examples 1 and 2, algorithms **MIFSFS** and **MIBSFS** always give a correct feature ranking, and also give the optimal best feature space as well as often indicating the irrelevance or redundancy of features. But algorithms **MIFS** and **MIFS_U** give a wrong feature ranking for examples 2 and 1, respectively.

5. Discussion and conclusions

From the above results on the problematic examples previously published it is clear that the proposed enhancement to MI-based feature selection is an improvement over the earlier techniques. This is to be expected as our technique assesses feature subsets in the context of feature interaction and the unavoidable extra computational cost appears to be non-prohibitive.

The stepwise basis for our survey algorithms means that the optimal feature subset may not be found. But a further unavoidable truth is that heuristic shortcuts must be used (as an exhaustive assessment of all possible feature combinations will, in general, be impractical), and the present results (as well as other not presented) suggest that our algorithms offer a reasonable compromise between suboptimality and practical efficiency. This is particularly so because we treat the first stepwise surveys backwards and forwards through the complete feature set as a basis for subsequent directed probes into the feature space when necessary.

Optimal input feature subset selection may not have a single optimal solution, as example 1, demonstrates, but by surveying the possibilities and then analysing the survey results a sensible selection may be made. It is the revelation of feature interactions that permits intelligent feature subset selection which may be preferable to blind algorithmic attempts to select an optimal subset. Ideally, feature irrelevance and redundancy will be signaled by a zero gain in MI when the irrelevant or redundant feature is added to a feature set. In addition, redundancy as opposed to irrelevance is indicated when variation in feature ranking is found. Thus in example 1, the survey algorithm results lead us to expect redundancy as either X^2Y and X , or X and Y appear to be optimal feature subsets. An inspection of pairwise single feature MI shows that the MI between X^2Y and X (approx. 0.5) and between X^2Y and Y (approx. 1.0) is an order of magnitude greater than between the pair X and Y (approx. 0.05). So as a result of our two MI surveys and a pairwise probe we know that there is considerable redundancy within these three features and we also know between which features it is concentrated. All of this is valuable knowledge when the detailed relationship between features is not known a priori, which is usually the case with practical application problems. By way of contrast the fixed rankings found with the Monk3 data suggest that the two features F_3 and F_6 are totally irrelevant (as their addition or deletion from feature sets does not change the MI value between the feature set and the class, see Tables 4 and 5). Feature F_1 is slightly more problematic as it sometimes appears irrelevant (e.g. Table 4) and sometimes appears to contain discriminatory information (e.g. Table 5). But as this information gain is small it is difficult to distinguish from noise, and because it is sometimes zero we would propose

that the occasional small positive gain is in fact noise. The complexity of the problem of distinguishing small discriminatory contributions from noise is exemplified in example 1 where no feature is totally irrelevant but certain features become effectively irrelevant with respect to certain feature subsets. A second problematic aspect of our approach concerns the necessary discretization of continuous-valued features. As the second treatment of data set A in Table 3 illustrates, by using more bins to discretize a feature we also introduce a bias towards that feature (see how feature X^2Y loses its first choice status when its bin size is reduced from 10 to 8 while feature X keeps its 10 bins and becomes favoured as always first choice). Thus the feature discretization component of our approach needs further investigation.

In sum, we believe that the important and difficult problem of feature selection is best addressed via algorithmic surveys in conjunction with intelligent decision making once the feature interactions are understood, at least in part. Our MIFSFS and MIBSFS algorithms are a first attempt to provide the necessary preliminary survey procedures, procedures that take all feature interactions into account and yet are efficient enough for most practical purposes.

Acknowledgements

Support for this research and for Dr Cang has been provided by the Engineering and Physical Sciences Research Council of the UK under grant GR/M75143. We also thank Dr Kwak for replying to our queries concerning the example problems.

References

- [1] S. Theodoridis and K. Koutroubas, *Pattern Recognition*, Academic Press: San Diego, 1999.
- [2] P. Pudil, J. Novovicova, and J. Kittler, (1994) Floating search methods in feature selection, *Pattern Recognition Letters*, 15: 1119-1125.
- [3] A. Jain, and D. Zongker, (1997) D Feature selection: evaluation, application, and small sample performance, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(2), 153-158.
- [4] S. Cang and D. Partridge, (2002) Feature ranking and optimal subset selection using mutual information, Res. Rep. No. 403, Dept. of Computer Science, Exeter University.
- [5] R. Battiti, Using Mutual Information for Selecting Features in Supervised Neural Net Learning *IEEE Trans. Neural Networks*, vol. 5, pp.537-550, July 1994.
- [6] N. Kwak and C-H. Choi, Input Feature Selection for Classification Problems *IEEE Trans. Neural Networks*, vol. 13, pp 143-159, Jan. 2002.
- [7] S. Cang and D. Partridge, Determining the Number of Components in Mixture Models Using Williams' Statistical Test, Accepted by the 8th International Conference on Neural Information Processing, China, Nov. 2001.

Appendix: Discrete Mutual Information

Mutual information is a measure of the dependence between the random variables. It is always symmetric and non-negative. It is zero if and only if the variables are independent.

The mutual information between two discrete random variables $U=(u_1, u_2, \dots, u_k)$ and $V=(v_1, v_2, \dots, v_d)$ is defined as :

$$I(U, V) = \sum_u \sum_v p(u, v) \log \frac{p(u, v)}{p(u)p(v)} \quad (1)$$

where $p(u, v)$ is a joint density function and $p(u)$ and $p(v)$ are the marginal density functions.

The mutual information between one continuous $X=(x_1, x_2, \dots, x_d)$ and one discrete $U=(u_1, u_2, \dots, u_k)$ random variables, is defined as

$$I(X, U) = \sum_u \int_{-\infty}^{\infty} p(x, u) \log \frac{p(x, u)}{p(x)p(u)} dx \quad (2)$$

where $p(x, u)$ is a joint density function and $p(x)$ and $p(u)$ are the marginal density functions.

The mutual information between class $C=(c_1, c_2, \dots, c_k)$ and discrete features $V=(v_1, v_2, \dots, v_d)$ defined in (2) can be written as

$$I(V, C) = \sum_c p(c) \sum_v p(v/c) \log \frac{p(v/c)}{p(v)} = \sum_c p(c) \sum_v p(v/c) \log p(v/c) - \sum_v p(v) \log p(v) \quad (3)$$

From (2), we can measure the mutual information between class C and continuous feature space $X=(x_1, x_2, \dots, x_d)$

$$\begin{aligned} I(X, C) &= \sum_c p(c) \int_{-\infty}^{\infty} p(X|c) \log \frac{p(X|c)}{p(X)} dX \\ &= \sum_c p(c) \int_{-\infty}^{\infty} p(X|c) \log p(X|c) dX \\ &\quad - \int_{-\infty}^{\infty} p(X) \log p(X) dX \end{aligned} \quad (4)$$

Mutual Information Between the Feature Space and the Class

The mutual information between the discrete feature $V=(v_1, v_2, \dots, v_d)$ and the class $C=(c_1, c_2, \dots, c_k)$ can be calculated using (4). We rewrite (4) as the following

$$\begin{aligned} I(V, C) &= \sum_c p(c) \sum_v p(v/c) \log p(v/c) \\ &\quad - \sum_v p(v) \log p(v) \end{aligned} \quad (5)$$

(5) will be used to calculate the mutual information between the feature space and the class.

Ensembles in Practice: Prediction, Estimation, Multi-Feature and Noisy Data

Stefan ZEMKE

DSV, Stockholm University/KTH, Forum 100, 164 40 Kista, Sweden

Abstract. This paper addresses 4 practical ensemble applications: time series prediction, estimating accuracy, dealing with multiple feature and noisy data. The intent is to refer a practitioner to ensemble solutions exploiting the specificity of the application area.

1. Introduction

Recent years have seen a big interest in ensembles – putting several classifiers together to vote – and for a good reason. Even weak, by itself not so accurate classifiers can create an ensemble beating the best learning algorithms. Understanding, why and when this is possible, and what are the problems, can lead to even better ensemble use. Many learning algorithms incorporate voting. Neural networks apply weights to inputs and a nonlinear threshold function to summarize the ‘vote’. Nearest neighbor (kNN) searches for k prototypes for a classified case, and outputs the prototypes’ majority vote. If the definition of an ensemble allows that all members classify, but only one outputs, then also Inductive Logic Programming (ILP) is an example.

A classifier can be also put into an external ensemble. Methods, how to generate and put classifiers together have been prescribed reporting accuracy over that of the base classifiers. But this success and generality of ensemble use does not mean that there are no special cases benefiting from a problem-related approach. This might be especially important in extreme cases, e.g. when it is difficult to obtain above-random classifiers due to noise – ensemble will not help. In such cases, it takes more knowledge and experiments (ideally by others) to come up with a working solution, which probably involves more steps and ingenuity than a standard ensemble solution. This paper presents specialized ensembles in 4 areas: prediction, estimating accuracy, dealing with multiple feature and noisy data. The examples have been selected from many reviewed papers, with clarity and generality (within their area) of the solution in mind. The idea of this paper is to provide a problem-indexed reference to the existing work, rather than to detail it.

1.1. Why Ensembles Work

An ensemble outperforms the base classifier due to several reasons [11]. First, given limited training data, a learning algorithm can find several classifiers performing equally well. An ensemble minimizes the risk of selecting the wrong one, as more can be incorporated and averaged. Second, the learning algorithm’s outcome – a classifier – might be merely a local optimum in the algorithm’s search. Ensemble construction restarts the algorithm from different points avoiding the pitfall. Third, a number of even simple classifiers together can express more complex functions – better matching the data.

It is not difficult to make ensembles work – most of the methods to ‘disturb’ either the training data, or the classifier construction result in an ensemble performing better than a single classifier. Ensembles have improved classification results for data repositories, having as the base learner different algorithms, thus suggesting that the ensemble is behind the progress. Ensembles can be stuck on top of each other adding benefits, e.g. if boosting is good at improving accuracy, bagging – at reducing variance, bagging boosted classifiers, or the other way, may be good at both, as experienced [34]. Ensemble size is not crucial, even small benefits. Much of the reduction in error appears at 10 – 15 classifiers [6]. But AdaBoost and Arcing measurably improve their test-set error until around 25 classifiers [25] or more.

Increased computational cost is the first bad news. An ensemble of tens, perhaps hundreds of classifiers takes that much more to train, classify and store. This can be alleviated by simpler base classifiers – e.g. decision stubs instead of trees – and pruning, e.g. skipping low-weight members in a weighted ensemble [22]. Overfitting can result when an ensemble does not merely model training data from many (random) angles, but tries to fit its whims. Such a way to boost accuracy may work on noise-free data, but in the general case this is a recipe for overfitting [31]. Readability loss is another consequence of voting classifiers. Rule-based decisions trees and predicate-based ILP, having similar accuracy as e.g. neural networks (ANN) and kNN were favored in some areas because of human-understandable models. However, an ensemble of 100 such models – different to make the ensemble work and possibly weighted – blows the readability.

A note on vocabulary. *Bias* refers to the classification error part of the central tendency, or most frequent classification, of a learner when trained on different sets; *variance* – the error part of deviations from the central tendency [34]. *Stable* learning algorithms are not that sensitive to changes in the training set, include kNN, regression, Support Vector Machines (SVM), whereas decision trees, ILP, ANN are *unstable*. *Global* learning creates a model for the whole data, later used (the model, not data) to classify new instances, e.g. ANN, decision tree, whereas *local* algorithms refrain from creating such models, e.g. kNN, SVM. An overview of learning algorithms can be found [23].

2. Common Ensemble Solutions

For an ensemble to increase accuracy, the member classifiers need to have: 1) Independent, or better negatively correlated, errors [1], 2) Expected above-random accuracy. Ensemble methods usually do not check the assumptions, instead they prescribe how to generate compliant classifiers. In this sense, the methods are heuristics, found effective and scrutinized for various aspects, e.g. suitable learning algorithms, ensemble size, data requirements etc.

An ensemble explicitly fulfilling the assumptions might be even more effective: smaller – including only truly contributing classifiers, more accurate – taking validated above-random classifiers etc. Experiments confirm this [21]. However, even more advanced methods often refer to features of common ensembles. There are many ways to ensure ensemble’s classifier diversity, e.g. by changing training data or the classifier construction process – the most common methods described below.

2.1. Different Training Instances

Different subsets of the training set lead to different classifiers, especially if the learning algorithm is unstable. The subsets could be without or with replacement, i.e. allowing multiple copies of the same example. A common setting is a *bootstrap* – drawing as many elements as in the whole set, however with replacement, thus having on average 63.2% elements of the original set, some repeated. Another possibility is to divide the training set into N disjoint subsets and systematically train on $N - 1$ of them, leaving

each one out. More general way is to assign weights to the examples and change them before training a new classifier respecting the weights.

Bagging [6] – classifiers trained on different bootstrap samples are put to a majority vote – class issued by most wins. Bagging improves accuracy by promoting the average classification of the ensemble, thus reducing influence of individual variances [13], with bias mostly intact [34], handles noise (to 20%) well [10], but does not at all work with stable learning methods.

AdaBoost [15] forms a committee by applying a learning algorithm to the training set whose distribution is changed, after generating each classifier, as to stress frequently misclassified cases. While classifying, a member of the ensemble is weighted by a factor proportional to its accuracy. With little training data, AdaBoost performs better than Bagging [10], however it may deteriorate if there is insufficient training data relative to the complexity of the base classifiers or their training errors grow [29].

2.2. Feature Selection

Classifiers can be trained with different feature subsets. The selection can be random or premeditated, e.g. providing a classifier with a selection of informative, uncorrelated features. If all features are independent and important, the accuracy of the restricted (feature-subset) classifiers will decline, however putting them all together could still give a boost. Features can also be preprocessed presenting different views of the data to different classifiers.

2.3. Changing Output Classes

The output values can be assigned to 2 super-classes, e.g. A_1, B_1 – each covering several of the original class values – and then training a classifier with the super-class, then making another selection, A_2, B_2 and training next classifier etc. When classifying, all ensemble members issue their super-classifications and a count is made which of the original classes appears most frequently – the final output.

Error Correcting Output Coding (ECOC) [12] is a multi-class classification, where each class is encoded as a string of binary code-letters, a codeword. Given a test instance, each of its code-letters is predicted, and the class whose codeword has the smallest Hamming distance to the predicted codeword is assigned. By reducing bias and variance, ECOC boosts global learning algorithms, but not local [27] – in that case ECOC code-letter's classifiers can be differentiated by providing them with a subset of features. In data with few classes ($K < 6$), extending the codeword length yields increases error reduction.

2.4. Randomization

Randomization could be inserted at many points resulting in ensemble variety. Some training examples could be distorted, e.g. by adding 0-mean noise. Some class values could be randomized. The internal working of the learning algorithm could be altered, e.g. by choosing a random decision among the 3 best in decision tree build up.

Wagging [3], a variant of bagging, requires a base learner accepting training set weights. Instead of bootstrap samples, wagging assigns random weights to instances in each training set, the original formulation used Gaussian noise to vary the weights.

2.5. Different Classifier Types

Even when trained on the same data, classifiers such as kNN, neural network, decision tree create models classifying new instances differently due to different internal language, biases, sensitivity to noise etc. Learners could also induce varied models due to different settings, e.g. network architecture.

Bayesian ensemble uses k classifiers obtained by any means in the Bayes formula. The basis for the ensemble outcome are probabilities: classes' and conditional for predicted/actual class pairs, for each classifier. The Bayes output, given k classifications, is then given by $class_p$ maximizing $P(class_p) * P_1(predict_1|class_p) * \dots * P_k(predict_k|class_p)$. It can be viewed as the Naive Bayes Classifier [23] meta-applied to the ensemble.

3. Specialized Ensemble Use

The quest for the ultimate ensemble technique resembles the previous efforts to find the 'best' learning algorithm which discovered a number of similarly accurate methods, some somehow better in specific circumstances, and usually further improved by problem-specific knowledge. Ensemble methods also show their strengths in different circumstances, e.g. no/data noise, un/stable learner etc. Problem specifics could be directly incorporated into an specialized ensemble. This section addresses four practical problem areas and presents ensemble adaptations. Though other problems in the areas might require individual approach, the intention is to bring some issues and worked-out solutions.

3.1. Time Series Prediction

Time series arise in any context in which data is linearly ordered, e.g. by time or distance. The index increment may be constant, e.g. 1 day, or not, as in the case of event-driven measurements, e.g. indicating a transaction time and its value. Series values are usually numeric, in a more general case – vectors of fixed length. Time series prediction is to estimate a future value, given values up to date. There are different measures of success, the most common accuracy – in the case of nominal series values, and squared mean error – in the case of numeric.

Series to instances conversion is required by most learning algorithms expecting as an input a fixed length vector. It can be a lag vector derived from series, a basic technique in nonlinear analysis $\mathbf{v}_t = (series_t, series_{t-lag}, \dots, series_{t-(D-1)*lag})$. Such vectors with the same time index t – coming from all input series – appended give an instance, its coordinates referred to as features. The lag vectors have motivation in Takens embedding theorem: [20] stating that a deterministic – i.e. to some extent predictable – series' dynamics is mimicked by the dynamics of the lag vectors, so e.g. if a series has a cycle – coming to the same values, the lag vectors will have a cycle too.

Embedding dimension D – the number of lagged series values used to model a series trajectory – according to the embedding theorem does not need to exceed $2d - 1$, where d is the dimension of the series generating system. In practice d is unknown, possibly infinite if the data is stochastic. D is usually arrived at by increasing the value until some measure – e.g. prediction accuracy – gets saturated. In theory – infinite data and no noise – it should stay the same even when D is increased, in practice it is not, due to curse of dimensionality etc. A smaller dimension allows more and closer neighborhood matches. An ensemble involving different dimensions could resolve the dilemma.

Embedding lag according to Takens theorem should only be different from the system's cycle, in practice it is more restricted. Too small, makes differences between the lagged values not informative enough to model the system's trajectory – imagine a yearly cycle by giving just several values separated seconds apart. Too big, misses the details and risks putting together weakly related values – as in the case of a yearly cycle sampled at 123-months interval. Without advanced knowledge of the data, a lag is preferred: either as the first zero autocorrelation or minimum of the mutual information [20]. However, those are only heuristics and an ensemble could explore a range of values, especially as theory does not favor any.

Prediction horizon – how much ahead to predict at a time – is another decision. Target 10 steps ahead prediction can be in 1 shot, in 2 iterated 5-ahead predictions, 5*2-ahead, or 10*1. Longer horizon makes the predicted quantity less corrupted by noise; shorter – can be all that can be predicted, and iterated predictions can be corrected for their systematic errors as described below. An ensemble of different horizons could not only limit outliers, but also estimate the overall prediction reliability via agreement among the individual predictions.

Converting a short-term predictor into longer-term can be also done utilizing some form of metalearning/ensemble [19]. The method uses a second learner to discover the systematic errors in the (not necessarily very good, but above-average) short-term predictor, as it is iterated. These corrections are then used when a longer-term prediction is issued resulting in much better results. The technique also provides an indication of a feasible prediction horizon and is robust w.r.t. noisy series.

Series preprocessing – meaning global data preparation before it is used for classification/prediction – can introduce domain-specific data view, reduce noise, normalize, presenting the learning algorithm with more accessible data. E.g. in the analysis of financial series, so called indicators series are frequently preprocessed/derived and consist of different moving averages and relative value measures within an interval [36]. Preprocessing can precede, or be done at the learning time, e.g. as calls to background predicates.

The following system [16] introduces general time series preprocessing predicates: **relative_increases**, **decreases**, **stays** (within a range) and region: **always**, **sometime**, **true_percentage** – testing if interval values belong to a range. The predicates, filled with values specifying the intervals and ranges, are the basis of simple classifiers – consisting of only one predicate. The classifiers are then subject to boosting up to 100 iterations. The results are good, though noisy data causes some problems.

Initial conditions of the learning algorithm can differ for each ensemble member. Usually, the learning algorithm has some settings, other than input/output data features etc. In the case of ANN, it is the initial weights, architecture, learning speed, weight decay rate etc. For an ILP system – the background predicates, allowed complexity of clauses. For kNN – the k parameter and weighting of the k neighbors w.r.t. distance: equal, linear, exponential. All can be varied.

An ANN example of different weight initialization for time series prediction follows [24]. Nets of the same architecture are randomly initialized and assigned to ensembles built at 2 levels. First, the nets are grouped into ensembles of fixed size Q , and the results for the groups averaged at the second level. Initially, $Q = 1$, which as Q increases expectably reduces the variance. At $Q = 20$ the variance is similar to what could be extrapolated for $Q = \infty$. Except for suggesting a way to improve predictions, the study offers some interesting observations. First, the minimum of the ensemble predictor error is obtained at ANN epoch that for a single net would already mean overfitting. Second, as Q increases, the test set error curves w.r.t. epochs/training time, go flatter making it less crucial to stop training at the 'right' moment.

Different series involved in a prediction of a given one, are another ensemble possibility. They might be other series than the one predicted, but supporting its prediction, what could be revealed by, e.g., significant non-zero correlation. Or the additional series could be derived from the given one(s), e.g. according to the indicator formulae, in financial prediction. Then all the series, can be put together into the lag vectors – already described for one series – and presented to the learning algorithm. Different ensemble members can be provided with their different selection/preprocessing combination.

Selection of delay vector lag, dimension, even for more input series, can be done with the following [35]. For each series, lag is set to a small value, and dimension to a reasonable value, e.g. 2 and 10. Next, a binary vector, as long as the sum of embedding dimensions for all series, is optimized by a Genetic Algorithm (GA). The vector, by its

'1' positions indicates which lagged values should be used, their number restricted to avoid the curse of dimensionality. The selected features are used to train a predictor which performance/accuracy measures the vector's fitness. In the GA population no 2 identical vectors are allowed and, after a certain number of generations, the top performing half of the last population is subject to majority vote/averaging of their predictions.

3.2. Multiple Features

Multiple features, running into hundreds or even thousands, naturally appear in some domains. In text classification, a word's presence may be considered a feature, in image recognition – a pixel's value, in chemical design – a component's presence and activity, or in a joint data base the features may mount. Feature selection and extraction are main dimensionality reduction schemes. In selection, a criterion, e.g. correlation, decides feature choice for classification. Feature extraction, e.g. Principal Component Analysis (PCA), reduces dimensionality by creating new features. Sometimes, it is impossible to find an optimal feature set, when several sets perform similarly. Because different feature sets represent different data views, simultaneous use of them can lead to a better classification.

Simultaneous use of different feature sets usually lumps feature vectors together into a single composite vector. Although there are several methods to form the vector, the use of such joint feature set may result in the following problems: 1) Curse of dimensionality, the dimension of a composite feature vector becomes much higher than any of component feature vectors, 2) Difficulty in formation, it is often difficult to lump several different feature vectors together due to their diversified forms, 3) Redundancy, the component feature vectors are usually not independent of each other [8]. The problems of relevant feature and example selection are interconnected [5].

Random feature selection for each ensemble classifier is perhaps the simplest method. It works if 1) data is highly redundant – it does not matter much which features are included, as many carry similar information and 2) the selected subsets are big enough to create above-random classifier – finding that size may require some experimentation. Provided that, one may obtain better classifiers in random subspaces than in the original feature space, even before the ensemble application. In a successful experiment [30], the original dimensionality was 80 (actually 24-60), subspaces – 10, randomly selected for 100-classifier majority vote.

Feature synthesis creates new features, exposing important data characteristics to classifiers. Different feature preprocessing for different classifiers ensures their variety for an effective ensemble. PCA – creating orthogonal combinations of features, maximizing variance – is a common way to deal with multi-dimensional data. PCA new features, principal components, generated in a sequence with decreasing variability/importance, in different subsets or derived from different data, can be the basis of an ensemble.

In an experiment to automatically recognize volcanos in Mars satellite images [2], PCA has been applied to $15 * 15$ pixels = 225 feature images. Varying number, 6 – 16, of principal components, plus domain features – line filter values – have been fed into 48 ANNs, making an ensemble reaching experts' accuracy. The authors conclude that the domain-specific features and PCA preprocessing were far more important than the learning algorithm choice. Such scheme seems suitable for cases when domain-specific features can be identified and the detection which other features contributed most is not important, since PCA mixes them all.

Ensemble-based feature selection reduces data dimensionality by observing which classifiers – based on which features – perform well. The features can then contribute to an even more robust ensemble. Sensitivity of a feature is defined as the change in the output variable when an input feature is changed within its allowable range (while holding all other inputs frozen at their median/average value) [14].

In in-silico drug design with QSAR, 100-1000 dependent features and only 50-100 instances present related challenges: how to avoid curse of dimensionality, and how to maximize classification accuracy given the few instances yet many features. A solution reported is to bootstrap an (ANN) ensemble on all features adding one random – with values uniformly distributed – to estimate sensitivities of features, and skip features less sensitive than the random. Repeat the process until not further feature can be dropped and train the final ensemble. This scheme allows to identify important features.

Class-aware feature selection – input decimation – is based on the following. 1) Class is important for feature selection (but ignored, e.g. in PCA). 2) Different classes have different sets of informative features. 3) Retaining original features is more human-readable. Input decimation works as follows [26]. For each among L classes, decimation selects a subset of features most correlated to the class and trains a separate classifier on that features. The L classifiers constitute an ensemble. Given a new instance, each of the classifiers is applied (to its respective features) and the class voted for by most is the output. Decimation reduces classification error up to 90% over single classifiers and ensembles trained on all features, as well as ensembles trained on principal components. Ensemble methods such as bagging, boosting and stacking can be used in conjunction with decimation.

3.3. Accuracy Estimation

For many real-life problems, perfect classification is not possible. In addition to fundamental limits to classification accuracy arising from overlapping class densities, errors arise because of deficiencies in the classifier and the training data. Classifier related problems such as incorrect structural model, parameters, or learning regime may be overcome by changing or improving the classifier. However, errors caused by the data (finite training sets, mislabelled patterns) cannot be corrected during the classification stage. It is therefore important not only to design a good classifier, but also to estimate limits to achievable classification rates. Such estimates determine whether it is worthwhile to pursue (alternative) classification schemes.

The Bayes error provides the lowest achievable error for a given classification problem. A simple Bayes error upper bound is provided by the Mahalanobis distance, however, it is not tight – might be twice the actual error. The Bhattacharyya distance provides a better range estimate, but it requires knowledge of the class densities. The Chernoff bound tightens Bhattacharyya upper estimate but it is seldom used since difficult to compute [33]. The Bayes error can be also estimated non-parametrically from errors of a nearest neighbor classifier, provided the training data is large, otherwise the asymptotic analysis might fail. Little work has been reported on a direct estimation of the performance of classifiers [4] and on data complexity analysis for optimal classifier combination [17].

Bayes error estimation via an ensemble [33] exploits that the error is only data dependent, thus the same for all classifiers that add to it extra error due to a specific classifier limitations. By determining the amount of improvement obtained from an ensemble, the Bayes error can be isolated. Given the error of a single classifier E , of an averaging ensemble $E_{ensemble}$, of N ρ -correlated classifiers, the Bayes error stands: $E_{Bayes} = \frac{NE_{ensemble} - ((N-1)\rho + 1)E}{(N-1)(1-\rho)}$. The classifier correlation ρ is estimated by deriving the (binary) misclassification vector for each classifier, and then averaging the vectors' correlations. This can cause problems, as it treats classifiers equally, and is expensive if their number, N , is high. The correlation can be, however, also derived via mutual information by averaging it between classifiers and an ensemble as a fraction of the total entropy in the individual classifiers [32]. This yields even better estimate of the Bayes error.

3.4. Noisy Data

There is little research specifically on ensembles for noisy data. This is an important combination, since most real-life data is noisy (in broad sense of missing and corrupted data) and ensembles' success may partially come from reducing influence of the noise by feature selection/preprocessing, bootstrap sampling etc.

Noise deteriorates weighted ensemble as the optimization of the combining weights overfits difficult, including noisy, examples [31]. This is perhaps the basic result to bear in mind while developing/applying elaborate (weighted) ensemble schemes. To assess the influence of noise, controlled amount of 5-30% input and output features have been corrupted in an experiment involving Bagging, AdaBoost, Simple and Arcing ensembles of decision trees or ANNs [25]. As the noise grew, the efficacy of the Simple and Bagging ensembles generally increased while the Arcing and AdaBoost gained much less. As for ensemble size, with its increase Bagging error rate did not increase, whereas AdaBoost's did.

Boosting in the presence of outliers can work, e.g., by allowing a fraction of examples to be misclassified, if this improves overall (ensemble) accuracy. An overview of boosting performance on noisy data can be found [18]. ν -Arc is AdaBoost algorithm enhanced by a free parameter determining the fraction of allowable errors [28]. In (toy) experiments, on noisy data ν -Arc performs significantly better than AdaBoost and comparably to SVM.

Coordinated ensemble specializes classifiers on different data aspects, e.g. so classifiers appropriately miss-classify outliers coming into their area, without the need to recognize outliers globally. Negatively correlated classifiers – making different (if need be) instead of independent errors – build highly performing ensembles [1]. This principle have been joint together with coordinated training specializing classifiers on different data parts [21]. The proposed ANN training rule – extension of backpropagation – clearly outperforms standard ANNs ensembles on noisy data both in terms of accuracy and ensemble size.

Missing data is another aspect of 'noise' where specialized ensemble solution can increase performance. Missing features can be viewed as data to be predicted – based on non-missing attributes. An approach sorts all data instances according to how many features they miss: complete instances, missing 1, 2, etc. features. A missing feature is a target for an ensemble trained on all instances where the feature is present. The feature is then predicted and the repaired instance added to the the data and the whole process repeated, if needed, for other features [9].

Removing mislabelled instances, with such cleaned data used for training, can improve accuracy. The problem is how to recognize a corrupted label, distinguishing it from exceptional, but correct, case. Interestingly, as opposed to labels, cleaning corrupted attributes may decrease accuracy if a classifier trained on the cleaned data later classify noisy instances. In an approach [7], all data has been divided into N parts and an ensemble trained (by whatever ensemble-generating method) on $N - 1$ parts, and used to classify the remaining part, in turn done for all parts. The voting method was consensus – only if the whole ensemble agreed on a class different from the actual, the instance was removed. Such a conservative approach is unlikely to remove correct labels, though may still leave some misclassifications. Experiments have shown that using the cleaned data for training the final classifier (of whatever type) increased accuracy for 20 - 40% noise (i.e. corrupted labels), and left it the same for noise less than 20%.

4. Conclusion

Ensemble techniques, bringing together multiple classifiers for increased accuracy, have been intensively researched in the last decade. Most of the papers either propose a

'novel' ensemble technique, often a hybrid one bringing features of several existing, or compare existing ensemble and classifier methods. This kind of presentation has 2 drawbacks. It is inaccessible to a practitioner, with a specific problem in mind, since the literature is ensemble method oriented, as opposed to problem oriented. It also gives the impression that there is the ultimate ensemble technique. Similar search for the ultimate machine learning proved fruitless. This paper concentrates on ensemble solutions in 4 problem areas: time series prediction, accuracy estimation, multiple feature and noisy data. Published systems, often blending internal ensemble working with some of the areas specific problems are presented easing the burden to reinvent them.

References

- [1] K. M. Ali and M. J. Pazzani. On the link between error correlation and error reduction in decision tree ensembles. Technical Report ICS-TR-95-38. Dept. of Information and Computer Science, UCI, USA, 1995.
- [2] L. Asker and R. Maclin. Feature engineering and classifier selection: A case study in Venusian volcano detection. In *Proc. 14th International Conference on Machine Learning*, pages 3–11. Morgan Kaufmann, 1997.
- [3] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting and variants. *To be published*, 1998.
- [4] H. Bensusan and A. Kalousis. Estimating the predictive accuracy of a classifier. Technical report, Department of Computer Science, University of Bristol, UK, 2001.
- [5] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, 1997.
- [6] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.
- [7] C. E. Brodley and M. A. Friedl. Identifying and eliminating mislabeled training instances. In *AAAI/IAAI, Vol. 1*, pages 799–805, 1996.
- [8] K. Chen and H. Chi. A method of combining multiple probabilistic classifiers through soft competition on different feature sets. *Neurocomputing*, 20:227–252, 1998.
- [9] C. Conversano and C. Cappelli. Incremental multiple imputation of missing data through ensemble of classifiers. Technical report, Department of Mathematics and Statistics, University of Naples Federico II, Italy, 2000.
- [10] T. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, ? :1–22, 1998.
- [11] T. Dietterich. Ensemble methods in machine learning. In *LNCS1857*, pages 1–15, 2000.
- [12] T. Dietterich and G. Bakiri. Error-correcting output codes: A general method of improving multiclass inductive learning programs. In *Proceedings of the Ninth National Conference on AI*, pages 572–577, 1991.
- [13] P. Domingos. Why bagging work? a bayesian account and its implications. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, pages 155–158, 1997.
- [14] M. Embrechts et al. Bagging neural network sensitivity analysis for feature reduction in qsar problems. In *Proceedings INNS-IEEE International Joint Conference on Neural Networks*, volume 4, pages 2478–2482, 2001.
- [15] Y. Freund and R. Schapire. A decision-theoretic generalization of online learning and an application to boosting. In *Proceedings of the Second European Conference on Machine Learning*, pages 23–37. Springer-Verlag, 1995.

- [16] C. A. Gonzalez and J. J. R. Diez. Time series classification by boosting interval based literals. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, 11:2–11, 2000.
- [17] T. K. Ho. Data complexity analysis for classifier combination. *Lecture Notes in Computer Science*, 2096:53–, 2001.
- [18] W. Jiang. Some theoretical aspects of boosting in the presence of noisy data. In *Proc. of ICML-2001*, 2001.
- [19] K. Judd and M. Small. Towards long-term prediction. *Physica D*, 136(1-2):31–44, 2000.
- [20] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis*. Cambridge Univ. Press, 1999.
- [21] Y. Liu and X. Yao. Negatively correlated neural networks for classification, 1998.
- [22] D. Margineantu and T. Dietterich. Pruning adaptive boosting. Technical report, Technical report: Oregon State University, 1997.
- [23] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.
- [24] U. Naftaly, N. Intrator, and D. Horn. Optimal ensemble averaging of neural networks. *Network*, 8(3):283–296, 1997.
- [25] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, pages 169–198, 1999.
- [26] N. C. Oza and K. Tumer. Dimensionality reduction through classifier ensembles. *Instance Selection: A Special Issue of the Data Mining and Knowledge Discovery Journal*, 2001.
- [27] F. Ricci and D. Aha. Error-correcting output codes for local learners. In *Proceedings of the 10th European Conference on Machine Learning*, 1998.
- [28] G. Rtsch, B. Scholkopf, A. Smola, K.-R. Mller, T. Onoda, and S. Mika. nu-arc: Ensemble learning in the presence of outliers, 2000.
- [29] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. In *Proc. 14th International Conference on Machine Learning*, pages 322–330. Morgan Kaufmann, 1997.
- [30] M. Skurichina and R. P. Duin. Bagging and the random subspace method for redundant feature spaces. In *Second International Workshop, MCS 2001*, volume 2096 of *LNCS*, 2001.
- [31] P. Sollich and A. Krogh. Learning with ensembles: How overfitting can be useful. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 190–196. The MIT Press, 1996.
- [32] K. Tumer, K. Bollacker, and J. Ghosh. A mutual information based ensemble method to estimate bayes error, 1998.
- [33] K. Tumer and J. Ghosh. Estimating the bayes error rate through classifier combining. In *International Conference on Pattern Recognition*, pages 695–699, 1996.
- [34] G. Webb. Multiboosting: A technique for combining boosting and wagging. Technical report, School of Computing and Mathematics, Deakin Univ., Geelong, Australia, 1998.
- [35] S. Zemke. Bagging imperfect predictors. In *Smart Engineering System Design*, volume 9, 1999.
- [36] S. Zemke. On developing a financial prediction system: Pitfalls and possibilities. In *ICML-02 Data Mining Lessons Learned*, 2002.

This page intentionally left blank

Section 8

Data Mining

This page intentionally left blank

Protoforms of Linguistic Data Summaries: Towards More General Natural-Language- Based Data Mining Tools

Janusz Kacprzyk and Sławomir Zadrozny
Systems Research Institute, Polish Academy of Sciences,
ul. Newelska 6, 01-447 Warsaw, Poland
E-mail: {kacprzyk,zadrozny}@ibspan.waw.pl

Abstract. We advocate the use of linguistic data summaries, exemplified by “most employees are young and well paid”, for a personnel database, as an intuitive and human consistent tool for data mining. We present an interactive approach, based on fuzzy logic and fuzzy database queries, which makes it possible to implement such summaries. We show how fuzzy queries are related to linguistic summaries, and show that one can introduce a hierarchy of prototype forms (protoforms), or abstract summaries in the sense of latest Zadeh's [1] ideas meant mainly for increasing deduction capabilities of search engines. For illustration we show an implementation for a sales database in a computer retailer, employing some type of a protoform of a linguistic summary.

1. Introduction

Data summarization is one of basic capabilities that is now needed by any “intelligent” system that is meant to operate in real life. Basically, due to the availability of relatively cheap and efficient hardware and software tools, we usually face an abundance of data that is beyond human cognitive and comprehension skills.

Since for a human being the only fully natural means of communication is natural language, a linguistic (say, by a sentence or a small number of sentences in a natural language) summarization of a set of data would be very desirable and human consistent. For instance, having a data set on employees, a statement (linguistic summary) like “almost all younger and well qualified employees are well paid” would be useful and human consistent.

This may clearly be an instance of a paradigm shift that is advocated in recent time the essence of which is to use natural language to represent values, relations, etc. characterizing some system or situation. This promises human consistency and simplicity though maybe at the expense of precision. This trend has become more and more pronounced recently, and a most prominent example is the so-called “computing with words paradigm” introduced by Zadeh in the mid-1990s, and extensively presented in Zadeh and Kacprzyk's [2] books.

Unfortunately, data summarization is still in general unsolved a problem in spite of vast research efforts. Very many techniques are available but they are not “intelligent enough”, and not human consistent, partly due to a little use of natural language. This concerns, e.g., summarizing statistics, exemplified by the average, median, minimum, maximum, α -

percentile, etc. which – in spite of recent efforts to soften them – are still far from being able to reflect a real human perception of their essence.

In this paper we will show the use of linguistic database summaries introduced by Yager [3, 4], and then considerably advanced by Kacprzyk [5], Kacprzyk and Yager [6], and Kacprzyk, Yager and Zadrożny [7, 8], and implemented in Kacprzyk and Zadrożny [9 - 15]. We will derive linguistic data summaries as linguistically quantified propositions as, e.g., “most of the employees are young and well paid”, with a degree of validity (truth, ...), in case of a personnel database.

We follow Yager's [3, 4] idea, and present its implementation, mainly using Kacprzyk and Yager [6], and Kacprzyk, Yager and Zadrożny's [7, 8] extensions. We employ Kacprzyk and Zadrożny's [9, 13] idea of an interactive approach to linguistic summaries in which the determination of a class of summaries of interest is done via Kacprzyk and Zadrożny's [17, 18] FQUERY for Access, a fuzzy querying add-on to Access, extended to the querying over the Internet in Kacprzyk and Zadrożny [19 - 21].

We show that by relating various types of linguistic summaries to fuzzy queries, with various known and sought elements, we can arrive at a hierarchy of prototypical forms, or – in Zadeh's [1] terminology – protoforms, of linguistic data summaries.

Finally, we present a real application for a sales database of a computer retailer.

2. Linguistic summaries using fuzzy logic with linguistic quantifiers

We follow Yager's [3, 4] concept of a linguistic summary of a set of data whose main elements are:

- V - a quality (attribute) of interest, e.g. salary in a database of workers,
- $Y = \{y_1, \dots, y_n\}$ - a set of objects (records) that manifest V , e.g. the set of workers; $V(y_i)$ are values of quality V for object y_i .

A *summary* of data set consists of:

- a summarizer S (e.g. young),
- a quantity in agreement Q (e.g. most),
- truth (validity) T - e.g. 0.7,

as, e.g., “ $T(\text{most of employees are young})=0.7$ ”.

Basically, the calculation of the truth (validity) of the basic type of a linguistic summary considered in this section is equivalent to the calculation of the truth value (from the unit interval) of a linguistically quantified statement (e.g., “most of the employees are young”). This may be done by using either Zadeh's [22] calculus of linguistically quantified statements (cf. Zadeh and Kacprzyk [23]) or Yager's [24] OWA operators [cf. Yager and Kacprzyk [25]].

A linguistically quantified proposition, exemplified by “most experts are convinced”, is written as “ Qy 's are F ” where Q is a linguistic quantifier (e.g., most), $Y = \{y\}$ is a set of objects (e.g., experts), and F is a (possibly)(usually?)fuzzy property (e.g., convinced). With importance added, I , we get “ QIy 's are F ”, e.g., “most (Q) of the important (I) experts (y 's) are convinced (F)”. We seek $\text{truth}(Qy \text{'s are } F)$ or $\text{truth}(QIy \text{'s are } F)$, respectively, knowing $\text{truth}(y \text{ is } F)$, $\forall y \in Y$.

Using Zadeh's [22] fuzzy-logic-based calculus of linguistically quantified propositions, property F and importance B are represented by fuzzy sets in Y , and a (proportional, nondecreasing) linguistic quantifier Q is assumed to be a fuzzy set in $[0, 1]$ as, e.g.

$$\mu_Q(x) = \begin{cases} 1 & \text{for } x \geq 0.8 \\ 2x - 0.6 & \text{for } 0.3 < x < 0.8 \\ 0 & \text{for } x \leq 0.3 \end{cases} \quad (1)$$

Then, due to Zadeh (1983)

$$\text{truth}(Qy's \text{ are } F) = \mu_Q[\frac{1}{n} \sum_{i=1}^n \mu_F(y_i)] \quad (2)$$

$$\begin{aligned} \text{truth}(QIy's \text{ are } F) = \\ = \mu_Q[\sum_{i=1}^n (\mu_I(y_i) \wedge \mu_F(y_i)) / \sum_{i=1}^n \mu_I(y_i)] \end{aligned} \quad (3)$$

An OWA operator (cf. [Yager [23], Yager and Kacprzyk [24]) of dimension p is a mapping $O: [0,1]^p \rightarrow [0,1]$ if with O is associated $W = [w_1, \dots, w_p]^T$, $w_1 + \dots + w_p = 1$, $w_i \in [0,1]$, a weighting vector, and

$$O(x_1, \dots, x_p) = w_1 b_1 + \dots + w_p b_p = W^T B \quad (4)$$

where b_i is the i -th largest element among x_1, \dots, x_p , $B = [b_1, \dots, b_p]$.

The OWA operators can model a wide array of aggregation operators (including linguistic quantifiers), from $w_1 = \dots = w_{p-1} = 0$ and $w_p = 1$ which corresponds to "all", to $w_1 = 1$ and $w_2 = \dots = w_p = 0$ which corresponds to "at least one", through all intermediate situations (corresponding to, e.g., linguistic quantifiers).

In the latter case the OWA weights may be found from the membership function of Q due to (cf. Yager [23]):

$$w_i = \begin{cases} \mu_Q(i) - \mu_Q(i-1) & \text{for } i = 2, \dots, p \\ \mu_Q(i) & \text{for } i = 1 \end{cases} \quad (5)$$

An important issue is related with the OWA operators for importance qualified data. Suppose that we have $A = [a_1, \dots, a_p]$, and a vector of importances $V = [v_1, \dots, v_p]$ such that $v_i \in [0,1]$ is the importance of a_i , $i = 1, \dots, p$, $v_1 + \dots + v_p = 1$. The case of an *ordered weighted averaging operator with importance qualification*, denoted OWA_Q is not trivial. In Yager's [23] approach, some redefinition of the OWA's weights w_i 's into \bar{w}_i 's is performed, and (4) becomes

$$O_I(x_1, \dots, x_p) = \bar{w}_1 b_1 + \dots + \bar{w}_p b_p = \bar{W}^T B \quad (6)$$

where

$$w_j = \mu_Q\left(\sum_{k=1}^j u_k / \sum_{k=1}^p u_k\right) - \mu_Q\left(\sum_{k=1}^{j-1} u_k / \sum_{k=1}^p u_k\right) \quad (7)$$

in which u_k is the importance of b_k , the k -largest element of A (i.e. the corresponding v_k).

Yager's [23] source approach to linguistic summaries is meant for one-attribute simple summarizers (concepts) - e.g., "young". Full search of the whole summaries' space is tacitly assumed. The simple form of the summarizer can be extended, for some confluence of attribute values as, e.g., "young and well paid", but this should be done "manually", and

implies combinatorial problems as a huge number of summaries should be generated and validated to find a most proper one. As in such a case the derivation of a proper linguistic summary requires the search of a huge amount of data, George and Srikanth [15] use for that purpose a genetic algorithm, and this approach is also considered here. However, we employ a different fitting function that involves, in addition to Yager's [2, 3] truth value (2) some other elements as, e.g.: a degree of imprecision (fuzziness), a degree of covering, a degree of appropriateness, a length of a summary, and yields an overall degree of validity.

For lack of space we refer the reader for details to Kacprzyk and Yager [6] or Kacprzyk, Yager and Zadrozny [7]. The overall quality (goodness) of a summary is given by the weighted sum of the above indicators, and the weights may be derived from expert testimonies from pair-wise comparisons between the particular indicators by using, e.g., Saaty's AHP (analytic hierarchy process) method.

Thus, basically, the problem is to find an optimal summary, $Su^* \in \text{SUMMARIES}$, such that

$$Su^* = \arg \max_{Su \in \text{SUMMARIES}} \sum_{i=1,2,\dots,5} w_i T_i(Su) \quad (8)$$

where w_1, \dots, w_5 , such that $\sum_{i=1,2,\dots,5} w_i = 1$, are weights assigned to the particular quality indicators, and T_i are these indicators.

3. A fuzzy querying add-on for formulating linguistic summaries

In Kacprzyk and Zadrozny's [9, 13] approach, *interactivity*, i.e. *user assistance*, is in the definition of summarizers (indication of attributes and their combinations). This proceeds via a user interface of a fuzzy querying add-on. Basically, the queries (referring to summarizers) allowed are:

- *simple* as, e.g., "salary is *high*"
- *compound* as, e.g., "salary is *low* AND age is *old*"
- *compound (with quantifier)*, as, e.g., "*most* of {salary is *high*, age is *young*, ..., training is *well above average*}."

We will also use "natural" linguistic terms, i.e. $(7 \pm 2)!$ exemplified by: *very low*, *low*, *medium*, *high*, *very high*, and also "comprehensible" fuzzy linguistic quantifiers as: *most*, *almost all*, ..., etc.

In Kacprzyk and Zadrozny [17, 18], a conventional DBMS is used, and a fuzzy querying tool is developed to allow for queries with fuzzy (linguistic) elements of the "simple", "compound" and "compound with quantifier" types. The main issues are: (1) how to extend the syntax and semantics of the query, and (2) how to provide an easy way of eliciting and manipulating those terms by the user.

FQUERY for Access is embedded in the native Access's environment as an add-on. All the code and data is put into a database file, a *library*, installed by the user. Definitions of attributes, fuzzy values etc. are maintained in a dictionary (a set of regular tables), and a mechanism for putting them into the Query-By-Example (QBE) sheet (grid) is provided. Linguistic terms are represented within a query as parameters, and a query transformation is performed to provide for their proper interpretation during the query execution.

It is obvious that fuzzy queries directly correspond to summarizers in linguistic summaries. Thus, the derivation of a linguistic summary may proceed in an interactive (user-assisted) way as follows:

- the user formulates a set of linguistic summaries of interest (relevance) using the fuzzy querying add-on described above,

- the system retrieves records from the database and calculates the validity of each summary adopted, and
- a most appropriate linguistic summary is chosen.

The use of fuzzy querying is very relevant because we can restate the summarization in the fuzzy querying context. First, (2) may be interpreted as:

"Most records match query S " (9)

where S replaces F in (2) since we refer here directly to the concept of a summarizer (of course, this should properly understood because S is in fact the whole condition, e.g., price = *high*, while S is just the fuzzy value, i.e. *high* in this condition; this should not lead to confusion).

Similarly, (3) may be interpreted as:

"Most records meeting conditions B match query S " (10)

Thus, (10) says something only about a subset of records taken into account by (9). In database terminology, B corresponds to a *filter* and (9) claims that *most* records passing through B match query S . Moreover, since the filter may be fuzzy, a record may pass through it to a degree from $[0,1]$.

It may be noticed that the concept of a protoform in the sense of Zadeh [1] is highly relevant in this context. First of all, a protoform is defined as an abstract prototype, that is, in our context, for the query (summary) given by (9) and (10) as follows, respectively:

"Most R 's are S " (11)

"Most BR 's are S " (12)

where R means "records", B is a filter, and S is a query.

Evidently, as protoforms may form a hierarchy, we can define higher level (more abstract) protoforms, for instance replacing *most* by a general linguistic quantifier Q , we obtain, respectively:

" QR 's are S " (13)

" QBR 's are S " (14)

Basically, the more abstract forms correspond to cases in which we assume less about summaries sought. There are two limit cases, where we: (1) assume totally abstract protoform or (2) assume all elements of a protoform are given on the lowest level of abstraction as specific linguistic terms. In case 1 data summarization will be extremely time-consuming, but may produce interesting, unexpected view on data. In case 2 the user has to guess a good candidate formula for summarization but the evaluation is fairly simple being equivalent to the answering of a (fuzzy) query. Thus, the second case refers to the summarization known as *ad hoc queries*. This may be illustratively shown in Table 1 in which 5 basic types of linguistic summaries are shown, corresponding to protoforms of a more and more abstracted form.

Table 1: Classification of linguistic summaries

Type	Given	Sought	Remarks
1	S	Q	Simple summaries through ad-hoc queries
2	$S\ B$	Q	Conditional summaries through ad-hoc queries
3	$Q\ S^{structure}$	S^{value}	Simple value oriented summaries
4	$Q\ S^{structure}\ B$	S^{value}	Conditional value oriented summaries
5	nothing	$S\ B\ Q$	General fuzzy rules

where $S^{structure}$ denotes that attributes and their connection in a summary are known, while S^{value} denotes a summarizer sought.

Type 1 may be easily produced by a simple extension of fuzzy querying as in FQUERY. Basically, the user has to construct a query – candidate summary, and has to be determined what is the fraction of rows matching this query and what linguistic quantifier best denotes this fraction. A Type 2 summary is a straightforward extension of Type 1 by adding a fuzzy filter. Type 3 summaries require much more effort. Their primary goal is to determine typical (exceptional) values of an attribute. So, query S consists of only one simple condition built of the attribute whose typical (exceptional) value is sought, the "=" relational operator and a placeholder for the value sought. For example, using the following summary in a context of personal data: $Q = \text{"most"}$ and $S = \text{"age=?"}$ (here "?" denotes a placeholder mentioned above) we look for a typical value of age. A Type 4 summary may produce typical (exceptional) values for some, possibly fuzzy, subset of rows. From the computational point of view. Type 5 summaries represent the most general form considered here: fuzzy rules describing dependencies between specific values of particular attributes. Here the use of B is essential, while previously it was optional. The summaries of Type 1 and 3 have been implemented as an extension to Kacprzyk and Zadrozny's [17 - 20] FQUERY for Access. Two approaches to Type 5 summaries producing has been proposed. Firstly, a subset of such summaries may be produced exploiting similarities with *association rules* concept and employing their efficient algorithms. Second, genetic algorithm may be employed to search the summaries' space. The results of the latter are briefly presented in the next section.

4. Implementation

As a simple illustration of Type 5 summaries, an implementation is shown for a sales database of a medium size computer retailer in Southern Poland. For illustration we will only show some examples of linguistic summaries for some interesting (for the user!) choices of relations between attributes.

First, discovered interesting relations between the commission and the type of goods sold are shown in Table 2. As we can see, the results can be very helpful in, e.g., negotiating commissions for various products sold.

Next, the relations between the groups of products and times of sale are shown in Table 2. Notice that in this case the summaries are much less obvious than in the former case expressing relations between the group of product and commission. It should also be noted that the weighted average is here very low but this, by technical reasons, should not be taken literally as these values are mostly used to order the summaries obtained.

Finally, let us show in Table 3 some of the obtained linguistic summaries expressing relations between the attributes: size of customer, regularity of customer (purchasing frequency), date of sale, time of sale, commission, group of product and day of sale. This is an example of the most sophisticated form of linguistic summaries accommodated in the system described.

Table 2. Linguistic summaries expressing relations between the group of products and commission

Summary	Degree of validity
About 1/2 of sales of network elements is with a high commission	0.3630
About 1/2 of sales of computers is with a medium commission	0.4753
Much sales of accessories is with a high commission	0.5713
Much sales of components is with a low commission	0.6707
About 1/2 of sales of software is with a low commission	0.4309
About 1/2 of sales of computers is with a low commission	0.4473
A few sales of components is without commission	0.0355
A few sales of computers is with a high commission	0.0314
Very few sales of printers is with a high commission	0.0509

Table 3. Linguistic summaries expressing relations between the groups of products and times of sale

Summary	Degree of validity
About 1/3 of sales of computers is by the end of year	0.2801
About 1/2 of sales in autumn is of accessories	0.4790
About 1/3 of sales of network elements is in the beginning of year	0.1957
Very few sales of network elements is by the end of year	0.0929
Very few sales of software is in the beginning of year	0.0958
About 1/2 of sales in the beginning of year is of accessories	0.4343
About 1/3 of sales in the summer is of accessories	0.3092
About 1/3 of sales of peripherals is in the spring period	0.2140
About 1/3 of sales of software is by the end of year	0.2258
About 1/3 of sales of network elements is in the spring period	0.2081
About 1/3 of sales in the summer period is of components	0.3081
Very few sales of network elements is in the autumn period	0.0955
A few sales of software is in the summer period	0.1765

Table 4. Linguistic summaries expressing relations between the attributes: size of customer, regularity of customer (purchasing frequency), date of sale, time of sale, commission, group of product and day of sale

Summary	Degree of validity
Much sales on Saturday is about noon with a low commission	0.3951
Much sales on Saturday is about noon for bigger customers	0.4430
Much sales on Saturday is about noon	0.4654
Much sales on Saturday is about noon for regular customers	0.4153
A few sales for regular customers is with a low commission	0.1578
A few sales for small customers is with a low commission	0.1915
A few sales for one-time customers is with a low commission	0.1726
Much sales for small customers is for non-regular customers	0.5105

These are the most valid summaries, and they give the user much inside into relations between the attributes chosen, moreover they are simple and human consistent.

5. Concluding remarks

We have presented an interactive, fuzzy logic based approach to the linguistic summarization of databases, and advocated it as a means to obtain human consistent summaries of (large) sets of data that can be easily comprehensible by human beings, hence that can be useful. We have indicated that Zadeh's [1] concept of protoforms, or a hierarchy of them, is a natural framework within which to consider various classes of linguistic summaries, and that this is particularly clear when we consider linguistic summaries from the perspective of their related fuzzy queries. The approach presented is implementable, and is a step towards an increase of the role of natural language in data mining.

References

1. Zadeh L.A.: A prototype-centered approach to adding deduction capabilities to search engines – the concept of a protoform. BISC Seminar, 2002), University of California, Berkeley, 2002.
2. Zadeh L.A., J. Kacprzyk J., Eds.: *Computing with Words in Information/Intelligent Systems*. Vol 1: Foundations, Vol. 2: Applications, Physica-Verlag (Springer-Verlag), Heidelberg and New York, 1999.
3. Yager R.R.: A new approach to the summarization of data. *Information Sciences*, vol. 28, 1982, pp. 69-86.
4. Yager R.R.: On linguistic summaries of data. In W. Frawley and G. Piatetsky-Shapiro (Eds.): *Knowledge Discovery in Databases*. AAAI/MIT Press, pp. 347-363.
5. Kacprzyk J.: Intelligent data analysis via linguistic data summaries: a fuzzy logic approach. In R. Decker and W. Gaul (Eds.): *Classification and Information Processing at the Turn of the Millennium*. Springer-Verlag, Berlin, Heidelberg and New York, 2000. pp. 153-161.
6. Kacprzyk J. and Yager R.R.: Linguistic summaries of data using fuzzy logic, *Int. J. of General Systems*, vol. 30. 2001, pp. 133-154.
7. Kacprzyk J., Yager R.R.. and Zadrozny S.: A fuzzy logic based approach to linguistic summaries of databases", *Int. J. of Applied Mathematics and Computer Science*, vol. 10. 2000. pp. 813-834.
8. Kacprzyk J., Yager R.R. and Zadrozny S.: Fuzzy linguistic summaries of databases for an efficient business data analysis and decision support. In W. Abramowicz and J. Zurada (Eds.): *Knowledge Discovery for Business Information Systems*. Kluwer, Boston, 2001, pp. 129-152.
9. Kacprzyk J. and Zadrozny S.: Data Mining via Linguistic Summaries of Data: An Interactive Approach. In T. Yamakawa and G. Matsumoto (Eds.): *Methodologies for the Conception, Design and Application of Soft Computing*" (Proc. of IIZUKA'98), Iizuka, Japan, 1998, pp. 668-671.
10. Kacprzyk J. and Zadrozny S.: On combining intelligent querying and data mining using fuzzy logic concepts. In G. Bordogna and G. Pasi (Eds.): *Recent Research Issues on the Management of Fuzziness in Databases*. Physica - Verlag, Heidelberg and New York, 2000. pp. 67-81.
11. Kacprzyk J. and Zadrozny S.: On a fuzzy querying and data mining interface, *Kybernetika*, vol. 36, 2000. pp. 657-670.
12. Kacprzyk J. and Zadrozny S.: Computing with words in intelligent database querying: standalone and Internet-based applications. *Information Sciences*, vol. 134, 2001, pp. 71-09.

13. Kacprzyk J. and Zadrożny S.: Data mining via linguistic summaries of databases: an interactive approach. In L. Ding (Ed.): *A New Paradigm of Knowledge Engineering by Soft Computing*. World Scientific, Singapore, 2001, pp. 325-345.
14. Kacprzyk J. and Zadrożny S.: On interactive linguistic summarization of databases via a fuzzy-logic-based querying add-on to Microsoft Access, In B. Reusch (Ed.): *Computational Intelligence – Theory and Applications*. Springer, Berlin, 1999, pp. 462-472.
15. Kacprzyk J. and Zadrożny S.: Fuzzy linguistic summaries via association rules. In A. Kandel, M. Last and H. Bunke (Eds.): *Data Mining and Computational Intelligence*. Physica-Verlag (Springer-Verlag), Heidelberg and New York 2001, pp. 115-139.
16. George R. and Srikanth R.: Data summarization using genetic algorithms and fuzzy logic. In F. Herrera and J.L. Verdegay (Eds.): *Genetic Algorithms and Soft Computing*. Physica-Verlag, Heidelberg, pp. 599-611, 1996.
17. Kacprzyk J. and Zadrożny S.: Fuzzy querying for Microsoft Access, *Proc. of FUZZ - IEEE'94* (Orlando, USA), 1994, vol. 1, pp. 167-171.
18. Kacprzyk J. and Zadrożny S.: FQUERY for Access: fuzzy querying for a Windows-based DBMS. In P. Bosc and J. Kacprzyk (Eds.): *Fuzziness in Database Management Systems*. Physica-Verlag (Springer-Verlag), Heidelberg, 1995, pp. 415-433.
19. Kacprzyk J. and Zadrożny S.: A fuzzy querying interface for a WWW-server-based relational DBMS. *Proc. of 6th IPMU Conference* (Granada), 1996, Vol. 1, pp. 19-24.
20. Kacprzyk J. and Zadrożny S.: Using fuzzy querying over the Internet to browse through information resources. In B. Reusch and K.-H. Temme (Eds.): *Computational Intelligence in Theory and Practice*. Physica-Verlag (Springer- Verlag), Heidelberg and New York, 2001, pp. 235-262.
21. Kacprzyk J. and Zadrożny S.: Data mining via fuzzy querying over the Internet. In O. Pons, M.A. Vila, J. Kacprzyk (Eds.): *Knowledge Management in Fuzzy Databases*. Physica - Verlag, Heidelberg, New York 2000. pp. 211-233.
22. Zadeh L.A.: A computational approach to fuzzy quantifiers in natural languages. *Computers and Maths with Appls.* Vol. 9, 1983, pp. 149-184.
23. Zadeh L.A. and Kacprzyk J., Eds.: *Fuzzy Logic for the Management of Uncertainty*, Wiley, New York, 1992.
24. Yager R.R.: On ordered weighted averaging operators in multicriteria decision making. *IEEE Trans. on Systems, Man and Cybern.* SMC-18, 1988, 183-190.
25. Yager R.R. and Kacprzyk J. (Eds.): *The Ordered Weighted Averaging Operators: Theory and Applications*. Kluwer, Boston, 1997.

Sparse Distributed Memory with Adaptive Threshold

J. L. Aguilar and N. Perozo

CEMISID. Dpto. de Computación.

Facultad de Ingeniería. Universidad de los Andes.

Av. Tulio Febres. 5101. Mérida, Edo. Mérida-Venezuela

Telf: (58.74)440002 Fax:(58.74)402872

email: aguilar@ing.ula.ve

Abstract. Sparse Distributed Memory is a content addressable, associative memory technique which relies on close memory items tending to be clustered together, with some abstraction and blurring of details. This paper discusses the limitations of the original model. Then, we propose a method which improve Sparse Distributed Memory efficiency through an adaptive threshold. The results obtained are good and promising.

1. Introduction

The Sparse Distributed Memory (SDM) was developed by Pentti Kanerva [5], and it may be regarded either as an extension of a classical or as a special type of three layer feedforward neural network. We can simulated some of the human cognitive capabilities because it work in a similar way of the human memory (associative memory recognition, etc.). SDM implements transformation from logical space to physical space using distributed data storing. A value corresponding to a logical address is stored into many physical addresses. This way of storing is robust and not deterministic. In, general, the main properties are [5]: a physical address can keep several logical addresses, the memory capabilities are large, a memory cell is not addressed directly, and if logical addresses are partially damaged, we can get correct output data.

The SDM standard uses a fix threshold to active the memory cell [3, 4, 5]. That simplifies the system but gives poor results because some of the memory cells can't be reusable due to that a fix threshold only allow recovery the information previously learn. That limits the capabilities and fault tolerance of the system and some information storing can't be recovery (specifically, when the number of information store is large). In this paper we propose an adaptive threshold to solve this problem. We present some experiments with results, which are promising.

2. Sparse Distributed Memory

SDM is a model proposed by Karneva that can work as an auto-associative memory technique where the contents and the addresses are from the same space and used alternatively [4, 5]. The inner workings of SDM rely on large binary spaces. The dimension of the space determines how rich is each word. Another important factor is how many actual memory locations are there in the space. In general, when we use a SDM, the features are represented as one or more bits. Groups of features are concatenated to form a word that becomes a candidate for writing into SDM. When writing copy of this binary

string is placed in all close enough hard locations. When reading, a close enough cue would reach all close enough hard locations and get some sort of aggregate or average out of them. Reading is not always successful. Depending on the cue and the previously written information, among other factors, convergence or divergence during a reading may occur. If convergence occurs, the pooled word will be the closest match of the input reading cue. On the other hand, when divergence occurs, there is not relation-in general-between the input cue and what is retrieved from memory. The main aspects of SDM are [2, 3, 4, 5]:

- ❑ The SDM calculates Hamming distances between the reference address and each location address. For each distance, which is less or equal to a given radius, the corresponding location is selected.
- ❑ The memory is represented by $n*m$ counters (where n is number of locations and m is the input data length) instead of single-bit storage elements.
- ❑ Writing to the memory, instead of overwriting, is as follows:
 - ❑ If the i -bit of the input data is 1, the corresponding counters (counters in the selected locations (rows) and in the i -th columns) are incremented
 - ❑ If the i -bit of the input data is 0, the corresponding counters are decreased
- ❑ Reading (or recall) from the memory is similar:
 - ❑ The contents of the selected locations are summed columnwise.
 - ❑ Each sum is thresholded. If the sum is greater than or equal to the threshold value, the corresponding output bit is set to 1, in the opposite case it is cleared.

Some of the limitations of the SDM are [2, 3]:

- The addresses and the data vector must be coded in a binary alphabet. Because a lot of real problems using a different alphabet, we need a system to translate the pattern of real problems in a code to be used for the SDM.
- The standard model uses an uniform distribution at the level of the input addresses. In the reality, the input vectors are grouped in a set distributed in a large multidimensional space. For that, if the addresses are choose randomly like is proposed by Karneva, a large number of cells maybe can't activate and other ones would be activate a lot of time.
- If we save a lot of data in a SDM, we can overlap a lot of information. In this case the results of the output vector will be incorrect. Only, if we store several time the same information, the probability to recovery a correct information that we have stored can increase. The reason is because the standard SDM use a fix threshold to activate the cells. That gives poor performances. For this last case is that we propose our approach.

3. Our adaptive threshold Approach

The standard SDM has been modified in order to use an adaptive threshold. That is, the decoding of the addresses is not fix, and the threshold is modified according to the results of the recognition phase of previous learned patterns. If the recognition rate is bigger than 95% that means that the threshold must continue fix, otherwise we must decrease or increase the threshold according to the quality of the recognition of previous learned patterns and the number of input patterns. Once modified the threshold, the SDM relearns the set of pattern with the new threshold. In this way, we can filter the information and introduce them in the SDM in the correct places. The threshold will be modified according to the next conditions:

1. If the recognition rate (Tr) is bigger than 95% then:

$$\text{threshold}(t) = \text{threshold}(t-1) \quad (1)$$

For this case, the threshold is not modified for the recognition phase.

2. If the recognition rate (Tr) is smaller than 95% and the data quantity stored is smaller than (number of memory cells/2) then:

$$\text{threshold}(t) = \text{threshold}(t-1) - (100 - Tr) * \bullet \quad (2)$$

- \bullet : is the learning rate ($0 \leq \bullet \leq 1$). In our experiment we have used $\bullet = 0.1$ because with it we have obtained the best results, see [2] for more details.
- Where Tr is the recognition rate, which is calculated during the recognition phase as the average recognition rate:

$$Tr = \frac{\sum_{i=1}^N Tr(i)}{N}$$

- N : number of information to be recovered.

For this case, the initial threshold for the learning phase must be a large value. In our experiment, we use 5 (see [2] for more details).

3. If the recognition rate (Tr) is smaller than 95%, and the data quantity stored is bigger than (number of memory cells/ 2) then:

$$\text{threshold}(t) = \text{threshold}(t-1) + (100 - Tr) * \bullet \quad (3)$$

- For this case, the initial threshold of the learning phase must be a small value. In our experiment, we use 3 (see [2] for more details).

The equations (1) to (3) are used until to improve the recognition rate (Tr) of the previous learned pattern. Then, we restart the learning phase of the SDM with the new threshold.

4. Experiments

In our experiment we use an Applet in Java to simulate the SDM. In addition, we compare our results with [1, 4]. We use the next set of parameters: the size of the input and output binary vectors of the SDM, the number of memory cells (NLA), initial threshold and the number of input patterns (N).

4.1 First Experiment

In this experiment, NLA is 15, N is 5 and the size of the input and output binary vectors of the SDM is 10 bits. During the learning phase, we learned the next set of information with threshold=5: {0111110000, 0011111000, 0001111100, 0000111110, 1000001111}.

The recognition rate of these learned patterns, with threshold=5, is equal to 83% (see [2]). Because $Tr = 83 \% \leq 95 \%$, we recalculate the threshold according to the equation (2),

$$\text{threshold}(t) = \text{threshold}(t-1) - (100 - Tr) * \bullet = 5 - (100-83)*0.1 = 3,3 \approx 3.$$

In this case, the recognition rate, with threshold = 3, is equal to 88% (see [2]). Because $Tr = 88 \% \leq 95 \%$, we recalculate the threshold according to the equation (2),

$$\text{threshold}(t) = \text{threshold}(t-1) - (100 - Tr) * \bullet = 3 - (100-88)*0.1 = 1,8 \approx 2.$$

Now the recognition rate, with threshold = 2, is equal to 83%. Because $Tr = 83 \% < 88 \%$ with threshold = 3, the optimal value of the threshold is 3.

We relearn the SDM with the threshold=3 and we obtain a recognition rate of the previous learned patterns equals to 95%. If we introduce some noise during the recognition phase, we obtain the next results:

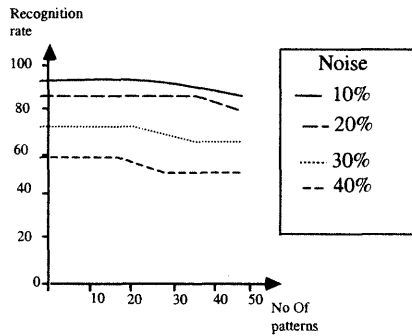


Figure 1. Recognition rate Vs noise

Now, in this phase we compare the performance of our approach with [1] and [4]. For each case, we have used 30 input patterns.

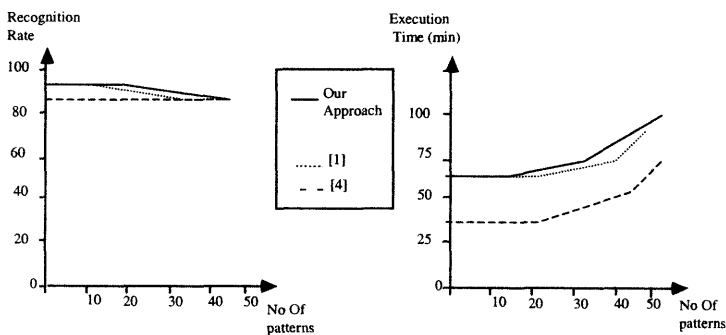


Figure 2. Comparison of the Recognition rate and execution time for a noise = 10%.

We improve the results obtained in [1, 4]. The reason is because our approach can adapt its procedure to the quality of the figures using the adaptive threshold. The execution time is very close to the evolutionary approach.

4.2 Second Experiment

With a number of input patterns (N) = 10, we use the next set of data during the learning phase with an initial threshold = 3: {0111110000, 0011111000, 0001111100, 0000111110, 0000011111, 1000001111, 1100000111, 1110000011, 1111000001, 1111100011}. The recognition rate of previous learned patterns, with threshold=3, is equal to 91% (see [2]). Because $Tr = 91 \% \leq 95 \%$, we recalculate the threshold using equation (3),

$$\text{threshold}(t) = \text{threshold}(t-1) + (100 - Tr) * \alpha = 3 + (100-91)*0.1=3,9 \approx 4$$

In this case, the recognition rate, with threshold = 4, is equal to 92% (see [2]). Because $Tr = 92 \% \leq 95 \%$, we recalculate the threshold using the equation (3),

$$\text{threshold}(t) = \text{threshold}(t-1) + (100 - Tr) * \alpha = 4 + (100-92)*0.1=4,8 \approx 5.$$

In this case, the recognition rate, with threshold = 5, is equal to 86% (see [2]). Because $Tr = 86 \% < 92 \%$ with threshold = 4, the optimal value of the threshold is 4.

We relearn the SDM with the threshold=4 and we obtain a recognition rate of the previous learned patterns equals to 93%. If we introduce some noise during the recognition phase, we obtain the next results:

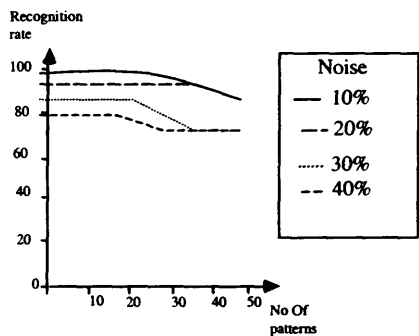


Figure 3. Recognition rate Vs noise

Now, in this phase we compare the performance of our approach with [1] and [4].

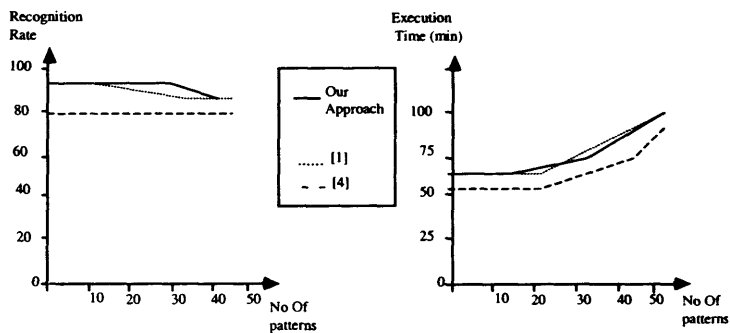


Figure 4. Comparison of the Recognition rate and execution time for a noise = 10%.

Similar to previous results, our approach improves the performance of previous work because we have obtained the optimal value of the threshold.

4.3 Third Experiment

In this part, we compare the performance of our approach for different number of bits of the input patterns. For this experiment and the next one, we use the best SDM model obtained to compare it with [1] and [4]. We suppose $NLA=15$ and $N=10$.

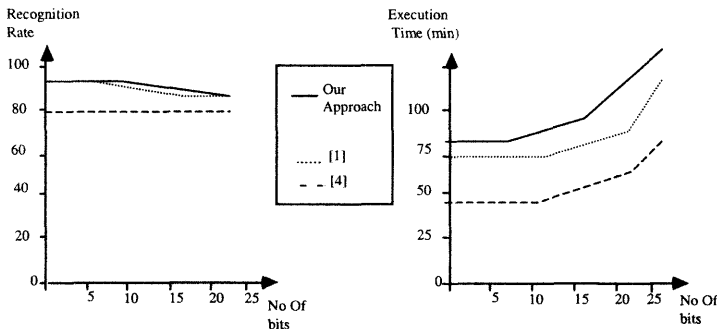


Figure 5. Comparison of the Recognition rate and execution time for a noise = 10% and different number of bits of the input patterns

In this case, the pattern size doesn't affect the performance of our approach. The problem is with our execution time, this increases when the number of bits is large.

4.4 Fourth Experiment

In this part, we compare the performance of our approach for different number of NLA. We suppose number of bits of the input patterns = 10 bits and $N=10$.

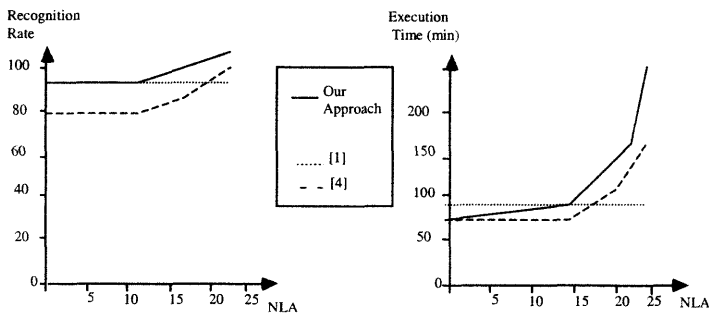


Figure 6. Comparison of the Recognition rate and execution time for a noise = 10% and different number of NLA

NLA improve the quality of the memory, because we increase the space to store the information and reduce the overlapping of the stored information. That mean, we improve the results quality, but with a large execution time.

4.5 Fifth Experiment

In this part, we compare the performance of our approach for different N . We suppose $NLA=15$ and number of bits of the input patterns = 10 bits.

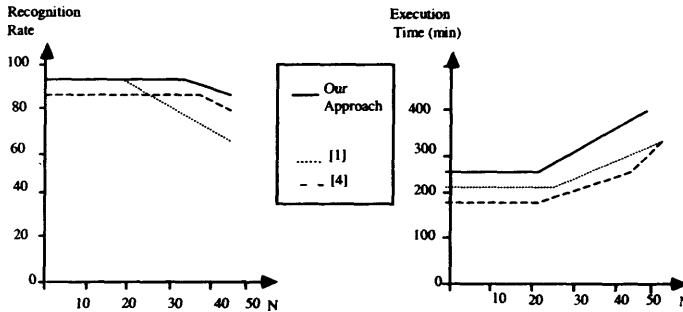


Figure 7. Comparison of the Recognition rate and execution time for a noise = 10% and different N

For this case, the execution time is the largest of all experiments because the learning phase needs more time to learn the set of patterns. In addition, the quality of solution is the worst of all experiments. The reason is because with a large N the information is overlapping. In general, [1] has problems to obtain good results when N is large. Otherwise, the SDM approaches performance decrease less with a large N .

5. Conclusions

The experimental results show that if we use an adaptive threshold in the SDM we can improve the performance of the recognition phase with respect to the classical SDM approach. In general, we obtain similar performance than the evolutionary approach proposed in [1]. But the recognition rate is the best for large N (one of the main capabilities of the SDM approaches is that). The execution time is increase because we relearn the SDM with the optimal value of threshold. We are going to extend our experiments for more complex problems (images recognition with a large number of bits, etc.).

Acknowledgment

This work was partially supported by FONACIT grant AP-97003817, CDCHT-ULA grant I-621-98-02-A.

References

- [1] J. Aguilar, A. Colmenares, Resolution of Pattern Recognition Problems using a Hybrid Genetic/Random Neural Network Learning Algorithm, *Pattern Analysis and Applications*, 1 (1998) 52-61.
- [2] J. Aguilar, Some experiments on the Sparse Distributed Memory, Technical Report 22-02, Cemisid, Universidad de los Andes, 2002.
- [3] F. Grebenicek, Data coding for SDM. Mosis'98 Proceedings, 1998, pp. 34-40.
- [4] F. Grebenicek, Self-Organized Sparse Distributed Memory- an Application: Pattern Data analysis. Technical Report 124, Ostrava University, 2000.
- [5] P. Kanerva, Sparse Distributed Memory. The MIT Press, Cambridge, Massachussets, 1990.

UniLR: An Automated Fuzzy Legal Reasoner

Dharmendra Sharma
*School of Computing,
University of Canberra, Australia*

Abstract

Automating legal reasoning is an interesting problem in artificial intelligence. In this paper, the problem of deciding on student disciplinary cases is studied, the various characteristics of the problem are identified and a prototype rule-based expert system that uses fuzzy reasoning is developed for the problem. The main motivation for the work is to study the difficulties in automating legal reasoning so that a sound verdict is reached for a small but an intricate domain. Some results are presented and the experience gained from the project is discussed. Future work is discussed on strengthening the prototype to include a formal case specification and an interaction language, and its drawing (through automated clustering) and use of relevant information from a base of previous cases. The UniLR prototype has been successfully tested for sample data.

1 Introduction

An expert system is a software that models human experts in some specific domain [2]. A typical expert system comprises of general knowledge base (facts and rules), case-specific data (clusters of information), inference engine, explanation subsystem, knowledge base editor and user interface. In real life, experts are mostly faced with challenges of making a decision 'on the fly'. In this case not all of the data needed to make a sound decision may be available, some may be suspect, and some of the knowledge for interpreting the data may be unreliable or themselves incomplete. The problem of drawing inferences from uncertain or incomplete data has confronted researchers and expert system developers giving rise to a variety of technical approaches. Some useful and typical mechanisms to deal with inexact reasoning involve approximate implication, possibility theory, certainty theory and fuzzy logic as discussed in Durkin [1], Hayes-Roth et al [3] and Negotia [7]. All these methods depend on the formalization of additional meta knowledge in order to correct the data, take back assumptions or combine evidence. The availability of this meta knowledge is a critical factor in the viability of these approaches to particular applications.

A number of legal expert systems attempt to implement various models of reasoning by simulating a lawyers approach to a legal problem (Popple [8]). In contrast, the practicality of a legal expert system very much depends on statistical approaches in reasoning which allows for quantification of vague concepts thereby attaching mathematical semantics to the decision making process.

This paper reports on a rule-based legal expert system shell (called UniLR) designed to use fuzzy reasoning in automated rulings on student discipline cases (on a motivating problem from the University of the South Pacific). UniLR is developed as a prototype to demonstrate the underlying concepts of the problem and is an attempt to support the decision-making process. It has an open architecture and designed in a general way so that

it can be used in like domains for legal reasoning. The motivation was to develop an automated reasoner to act as a decision support tool.

UniLR makes use of a knowledge base consisting of regulations and judgmental rules of the underlying legal system, derived from the knowledge engineering process applied on the domain. Ideally, any attempt to rid the vagueness of our natural language, using which the rules and regulations are usually expressed, is chimerical. Hence, UniLR uses fuzzy logic as a formal systematic approach to inexact reasoning to give judgement at the end of the case investigation and interrogation process. Its core functionality is to give ruling on charges triggered by a given case stated and justify its decision during the judgment delivery process.

2 Fuzzy Logic

The notion of 'fuzzy system' has evolved around human minds since 1920s. Zadeh [13] extended the work on *possibility theory* into a formal system of mathematical logic. Fuzzy Logic is a superset of conventional (boolean) logic that has been extended to handle the concept of partial truth values between "completely true" and "completely false". It uses degrees of membership in sets rather than a strict true/false membership. Fuzzy logic is primarily concerned with quantifying and reasoning about vague or fuzzy terms that appear in our natural language. In fuzzy logic, these fuzzy terms are referred to as linguistic or fuzzy variables. A fuzzy set assigns membership values between 0 and 1 that reflect more naturally a member association with the set. Let X be the range of possible values of fuzzy variable, with elements of X denoted as x . A fuzzy set A of X is characterized by a membership function $\mu_A(x)$ that associates each element x with a degree of membership value in A : $\mu_A(x): X \rightarrow [0,1]$. Event or element x is assigned a membership value by a membership function μ . The value represents the degree to which element x belongs to fuzzy set A : $\mu_A(x) = \text{Degree}(x \in A)$ where, $0 \leq \mu_A(x) \leq 1$. For a discrete set of elements, a fuzzy set can be represented through the use of a vector: $A = (a_1, a_2, \dots, a_n)$ where, $a_i = \mu_A(x_i)$. Fuzzy logic is increasingly used in artificial intelligence applications. (A good example is the Sony Palm Top that apparently uses a fuzzy logic decision tree algorithm to perform Kanji character recognition.) Most applications use it as the underlying logic system.

3 The legal problem

In general, there are two kinds of laws [9]: *Case law* (decisional) in which the decision is made in court, *Statute* (definitional) which is determined by the legislation as stated in Sergot et al.

UniLR is designed to cater for both kinds of laws. As a case study for UniLR prototype development, we use the University of the South Pacific Discipline Ordinance and Regulations for Students [12]. It contains both kinds of rules and regulations: definitional and decisional. The knowledge acquisition, interpretation and formalization has been done with some help from legal experts. Some sample regulations pertaining to discipline considered for UniLR are summarized below.

- Campus regulations – regulations to enforce discipline on campus. For example, campus regulation C14 states: *Students, whenever they are on campus, shall have in their possession at all times, a valid student ID card and shall produce the same to an officer of the University on Demand.*
- Residential regulations – regulation to see that discipline is maintained in Halls of Residence. For example, residential regulation R3 states: *Pets, animals or birds are not allowed in the Halls of Residence.*

- Assessment and examination regulations – regulations to see that honest practice prevail in assessment and examinations. For example, assessment and examination regulation 1.9 states: *No Candidate is to bring with him into the examination room any written or printed matter except: (a) as authorised by the examiner; or (b) where such written or printed material has been authorised for use in an approved open book examination.*

A legal case is declaratively expressed as a story in a given number n of sentences. The main objective is to prove the truth or falsity of each sentence in the story, which in turn determines the guilty status of filed charges. Each of the sentences in the case story has its own search space to its truth or falsity with some attributed degree of confidence. A case can have many charges, a charge can be proven to be guilty [G], not guilty [NG] or have insufficient evidence [NSE]. A sentence has *supports* as evidence and a set of *askables* for further interrogation of the user to elicit more information. The problem-solver is tasked with traversing a search space for a sentence to prove its truth or falsity.

4 UniLR Architecture

The UniLR shell consists of the functional components - a charge generator, an interrogator and a as described below.

1. Charge Generator:
 - Picks a case sentence.
 - If sentence is valid then produce charge by the violated regulation i.e. cluster the fact by violated regulation else write to error file. (This is because of the notion of closed world assumption – the system will not recognize any sentence that is not defined.)
 - Repeats the above process for each of the stated case facts.
2. Interrogator:
 - Picks a charge with its matching case facts.
 - Interacts with the user.
 - Uses offence justification rules and the Fuzzy Reasoner to determine whether guilty or not guilty.
3. Fuzzy Reasoner:
 - Provides reasoning service to the Interrogator based on the fuzzy logic system.
 - Stores the result of the current case in a file to support future learning using case based matching.
 - Displays the results at the end of the interrogation process.

Knowledge Base

UniLR's knowledge base consists of:

1. Case story (facts) – information on a given case.
2. Rules(coded) – expresses the rules and regulations of the underlying legal system and used to come up with charges during the charge generation process.
3. Judgmental rules – initiates expression of knowledge of a expert with the quest for evidence/support for the sentence/offence in consideration. Based on the result of search for support/evidence, these rules may trigger *askables* or other *jrules*.
4. Askables – are questions used to extract any additional information from the user.
5. Regulations (text) – specifies the text version of the regulations from the application domain.
6. Penalty – expresses the penalties assigned to each of the rules.
7. Previous case results - includes verdicts from previous cases.

Knowledge Representation

The UniLR shell currently uses knowledge represented as Prolog clauses. The *<keyword>* used in the knowledge base can be functors to attach additional semantics to a case fact or evidence. Specific format for each component of the knowledge base is summarized as follows.

- 1. Case story consist of the following first-order predicate clauses
 - student(<id>)** – specifies the student being accused
 - st(<stid>,<stkeyword>,'<sentence>')** – expresses each sentence in the case story that has violated a rule. The *<stkeyword>* should be found in one of the rules. *<sentence>* is the text version of the *<stkeyword>*.
 - su(<stid>,<sukeyword>)** – expresses support/evidence for each encoded sentence. *<stid>* specifies the link between a support and a sentence. *<sukeyword>* should be found in one of the rules. The use of *<id>* allows same support text to be used for different sentences.

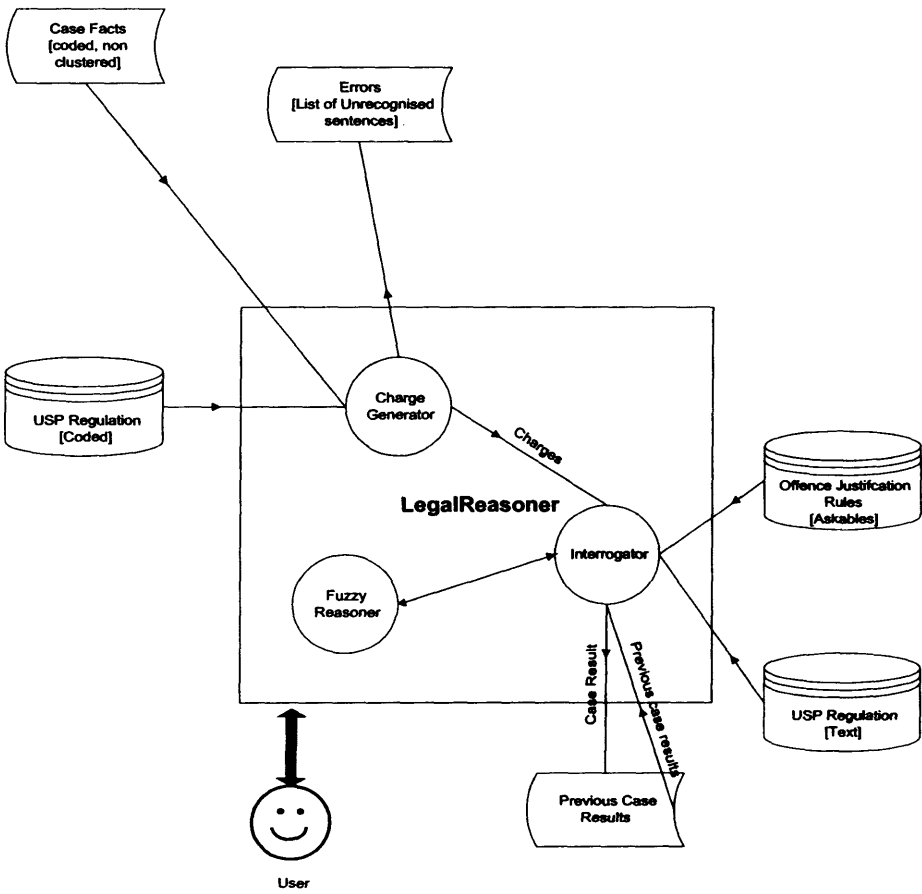


Figure 1: Architecture of UniLR

2. Rules

rule(<rule id>,<list of keywords>) – where the <list of keywords> should be used to express the case sentences.

3. Judgemental rules

jrule(<jrule id>, G, NG, NSE, ID, <sukeyword>, X, F) – where the <jrule id> uniquely identifies each of the judgmental rules. G, NG, NSE are the fuzzy variable which gets membership values assigned on the result of the quest for the desired evidence/support. ID identifies the sentence being processed, X caches the result of the evidence search, and F specifies link of this jrule to other jrules and/or askables.

4. Askables

ask(<ask id>,G , NG, NSE, A, F) - where the <ask id> uniquely identifies each of the askables. G, NG, NSE are fuzzy variables which gets membership values assigned on the response of the user. A caches the user response, and F specifies link of this askable to other jrules or askables

5. Regulations – text version of rules

regulation (<rule id>, ‘<regulation text>’) – where the <rule id> identifies its corresponding rule

6. Penalty

penalty(<rule id>, <offence number>, ‘<penalty text>’) – where the <rule id> identifies its corresponding rule

7. Case result – at the end of the case processing, the information of the entire case is captured as a single Prolog clause in the following format. This caters for future matching with previous cases, extending the UniLR to a hybrid of rule-based and case-based expert system shell.

**Case(<student id>, [[<rule id>, <AG>, <ANG>, <ANSE>, <list of sentences violating>, <rule id>], <list of supports and responses to askable>]
[<rule id> ,.....]......])**

where the <rule id> identifies a charge and <AG>, <ANG>, <ANSE> specify average membership values per charge to each of the fuzzy sets on which the ruling is based.

An element in the <list of sentences violating rule id> is the following format:

[<sentence id>,<stkeyword>,’<sentence>’]

An element in the <list of supports and responses> is in the following format:

[<guilty membership value> <not guilty membership value> <not sufficient membership values> <response/support id><response/sukeyword>]

~ in front of the sukeyword means support not found

Meta-knowledge

Some of the meta knowledge and rules used in UniLR shell are as follows:

1. Rule in which the defendant takes plea gets priority over other rules at any point in time. For example, at time *t* the conflict set contains rules R11, R12, R13 and out of which R12 is the rule which determines the plea of the defendant then R12 will take priority during conflict resolution.
2. Definitional rules do not require interrogation. For example, Examination Regulation 1.9 states that: *No Candidate is to bring with him into the examination room any written or printed matter except: (a) as authorised by the examiner; or (b) where such written or printed material has been authorised for use in an approved open book examination.* If a student is caught using unauthorised material in the exam then s/he is automatically guilty and the system can be used to find out the penalty guidelines.

3. Since each of the sentences has its own support and askables (i.e. a separate search space), the initial conflict set can be resolved in any order as all the applicable rules are valid at the initial problem solving point. The current version of the *UniLR* shell selects the first rule in the conflict set based on the default order of rules in the prolog database, in other words, the rules are loaded in an ordered fashion with respect to its importance to the problem, the shortest possible solution path. However, further conflict resolution should select first rule in the conflict set (list) as the fired rule changes the conflict set.
4. At the end of the interrogation, if decision is not clearer (low maximum membership value [$<6\%$] from the fuzzy reasoner) then the Rule to get the opinion of the user is fired. Depending on the response, this alters the decision state of the system before the rule was fired else the system adheres to its previous decision state.

Search

The search is primarily depth first with backtracking done during the case investigation process. It uses forward chaining through symbolic pattern matching. The shell is implemented in such a way that it contains two different phases of search and both of them are depth-first with backtracking. In the first level of search, the *UniLR* shell traverses through the encoded case story of n sentences $\langle S_1, S_2, S_3, \dots, S_n \rangle$ and the encoded regulation $\langle R_1, R_2, R_3, \dots, R_p \rangle$ generating m charges $\langle C_1, C_2, C_3, \dots, C_m \rangle$ for the case story where $m \leq n$. This search is very simple as it traverses through the search space and determines which regulations has been violated in the case story. In the second level of search, *UniLR* shell traverses the search space for each charge interrogating in a depth first manner. At each node of the search tree, the shell either quests for an evidence or fires an askable to get a judgmental decision from the user. The search tree for each of the charges under microscope is dynamic and is created during the interrogation process from the responses and evidence encountered. Based on the result of each node, the shell assigns a membership value to each of the fuzzy sets $\langle G, NG, NSE \rangle$. The shell traverses m sub trees to determine guilty status of each of the m charges.

Membership Value Assignment

The shell is very flexible in membership value assignment for the fuzzy variables. It allows membership values to be assigned during the knowledge engineering process. This caters for context dependency of the legal system being used. It also overcomes the problem of deep inferencing while using certainty factor as discussed in Durkin [1]. Further to this, it provides for integration of multiple experts' knowledge, through manipulation of membership values by the rules of fuzzy algebra. Rules of fuzzy algebra involves the notion of concentration, dilation, intensification, power, intersection, union, addition, subtraction, multiplication, division, and complementation of membership values and linguistic variables. Due to the nature of the problem and a generic approach in building the shell, the semantics of the particular problem solving has been taken into consideration. The membership values assigned to the fuzzy variables exhibiting the degree of belonging, is a variable for each interrogation depending on the context of the sentence involved. For example, a $\langle nq \rangle$ *not quite* response for question 1, an expert may mean weight of 0.3 to Fuzzy Set A . On the other hand, a $\langle nq \rangle$ *not quite* response for question 2, the expert may mean weight of 0.1 to Fuzzy Set A . This dilemma is conquered by the use of the mentioned knowledge representation format.

In contrast, with the utilization of the mentioned membership approach, the question of quantification of responses and evidence comes into picture. As the *UniLR* Shell needs the membership values to be coded during the knowledge engineering stage, a rule of thumb (heuristic) is used by the knowledge engineer to quantify its askables and evidence. For example, if *Guilty* fuzzy variable is assigned 0.8 for a response, then based on the nature of the question and response, the *Not Guilty* fuzzy variable could be assigned a value that is

the complement of the *Guilty* fuzzy variable. However, this varies from evidence to evidence and question to question. Hence a general heuristics for membership values assignment will have problems in reasoning the quantification of domain specific knowledge there by rising the complexity of knowledge engineering process. This calls for the heuristic to be response and evidence dependent, although similar responses can use same heuristic in membership values assignment to the fuzzy variable.

Vagueness and Fuzzy Reasoning

UniLR implements fuzzy reasoning in judgement delivery process. It uses a linguistic variable called *CHARGE*. *CHARGE* has three fuzzy sets, *GUILTY [G]*, *NOT GUILTY [NG]*, *NOT SUFFICIENT EVIDENCE [NSE]*, defined on it which captures partial membership of a response and/or evidence to the fuzzy variable. Hence, $\rightarrow \mu_A(x) = \text{Degree}(x \in A)$, where $0 \leq \mu_A(x) \leq 1$, A is G , NG and NSE , $X = \text{charge}$. At the end of the case investigation process the fuzzy reasoner does the average membership value, AMV_A , for each of the fuzzy set to come with a decision of AMV_A degrees of confidence. The highest AMV_A of each of the fuzzy sets determines the guilty status of the charge under the microscope.

$$MVA_A = \frac{\sum_{i=1}^n A_i}{n} \quad \text{where } A = G, NG, NSE \text{ and } n = \text{no. of evidence/askable for each } A.$$

UniLR Algorithm and descriptions

1. System Initialization
2. Charge Generator

```

BEGIN Charge Generation
  COLLECT all the sentence id's in list l
  WHILE list l is not empty
    GET the head of list l
    GET the corresponding sentence
    VALIDATE the sentence
    IF Valid THEN
      FIND the regulation violated by the sentence
      CLUSTER sentence by violated rule
      ;one rule can be violated by many sentences
    ELSE
      WRITE the sentence id to the error file
    END IF
  END WHILE
END

```

3. Interrogator

```

BEGIN Interrogation
  WHILE not end of charge list CL
    SELECT a charge C from CL
    WHILE not end of sentence list SL for CL
      SELECT a sentence S from SL
      ;SL is list of sentences violating the charged
      ;rule. For n charges there will be n SL of
      ;varying size.
      COLLECT all the applicable jrules giving AR
      ;AR is the conflict set on SL
      WHILE conflict set RA is not empty
        SELECT the first rule R in AR
        ;conflict resolution
      END WHILE
    END WHILE
  END WHILE

```

```

        FIRE rule R
        COLLECT response/evidence with assigned
        membership values of R
        IF R triggers other rules then
            PREPEND the new applicable rules to the
            conflict set RA
        END IF
        RA = RA - R
    END WHILE
    SL = SL - S
END WHILE
CL = CL - C
END WHILE
END

```

4. Fuzzy Reasoner

```

BEGIN fuzzy reasoning
    WHILE not end of charge list CL
        SELECT charge C from CL
        COLLECT all the membership value for guilty fuzzy
        variable G
        COMPUTE average of membership  $AMV_G$  for G
        STORE  $AMV_G$  with C
        COLLECT all the membership value for not guilty fuzzy
        variable NG
        COMPUTE average of membership  $AMV_{NG}$  for NG
        STORE  $AMV_{NG}$  with C
        COLLECT all the membership value for not sufficient
        fuzzy variable NSE
        COMPUTE average of membership  $AMV_{NSE}$  for NSE
        STORE  $AMV_{NSE}$  with C
        CL = CL - C
    END WHILE

    WHILE not end of charge list CL
        SELECT charge C from CL
        PERFORM fuzzy reasoning based on  $MVA_C$ 
        DISPLAY judgement
        DISPLAY justification of the judgement - I/O
        CL = CL - C
    END WHILE
END

```

5 Conclusion and Further Work

UniLR is a successful realization of fuzzy logic in a legal reasoning environment. In this paper, the legal reasoning problem pertaining to student discipline cases has been abstracted, conceptualized and a system developed on an architecture realizing the fuzzy reasoning metaphor. A prototype has been developed to exhibit the practicality of the researched concept at both abstract (conceptual) and implementation (concrete) level with respect to the problem nature/domain (legal case) and problem solving paradigm (fuzzy logic). Fuzzy reasoning has been successfully implemented in the shell. Although, the utilized rules of fuzzy algebra could change with respect to a particular application within the problem domain, the methodology of realizing fuzzy logic metaphor will remain the same. Further, the implemented knowledge representation format is rich enough to cater for easier extension of the shell from rule based to a hybrid of case based and rule based legal expert system shell. Also, the shell is generic in nature separating the problem solving engine from the problem and domain specific knowledge and hence, can be used with any legal system that supports decisional and/or definitional nature of cases.

Further work on the shell include development of problem specific heuristics for membership value assignment to the fuzzy variable; support for *why* question during the interrogation process; matching with previous case results either using pure symbolic pattern matching and/or values of fuzzy variables to incorporate learning into shell; and English like natural language user interface. An induction subsystem needs to be developed to learn from previous cases and use the knowledge to facilitate processing of new cases through pattern matching. A COBWEB system based on [2] is currently being investigated for the conceptual clustering of previous case data and its use in further strengthening UniLR.

References

1. Durkin J. 1994. *Expert Systems – Design and Development*. Macmillan Publishing Company. New York.
2. Fisher D. H. *Knowledge Acquisition via Incremental Conceptual Clustering*. In Machine Learning, pages 139-172, 1987.
3. Hayes-Roth F., Waterman D.A. and Lenat D. B. 1993. *Building Expert Systems. Vol. 1*. Addison-Wesley Publishing Company, Inc. Canada.
4. Le T. V. 1993. *Techniques of Prolog Programming*. John Wiley & Sons, Inc. Canada.
5. Mellish C. S. and Clocksin W. F. 1994. *Programming in Prolog. 4th Ed*. Springer-Verlag. New York.
6. Meritt D. 1989. *Building Expert Systems in Prolog*. Springer-Verlag. New York.
7. Negoita C. V. 1985. *Expert Systems and Fuzzy Systems*. The Benjamin/Cummings Publishing Company, Inc. California.
8. Popple J. 1996. *A Pragmatic Legal Expert System*. Aldershot Dartmouth. U.K.
9. Puppe F. 1993. *Systematic Introduction to Expert Systems – Knowledge representations and problem solving methods*. Springer-Verlag. New York.
10. Sergot M. J., Sadri F., Kowalski R. A., Kriwaczek F., Hammond P., and Cory H. T. 1986. *The British Nationality Act As A Logic Program*. Communications of the ACM. Vol. 29(5).
11. Shoham Y. 1994. *Artificial Intelligence Techniques in Prolog*. Morgan Kaufman Publishers, Inc. U.S.A.
12. University of the South Pacific Discipline Regulations, 1998.
13. Zadeh, L. A. *Fuzzy sets*. Information and Control, Vol 8, pp. 338-353.

Set Approximation Quality Measures in the Variable Precision Rough Set Model

Wojciech Ziarko

*Department of Computer Science
University of Regina, Regina, SK, S4S 0A2 Canada*

Abstract. The article introduces the basic notions of the variable precision rough set model (VPRS) including the parametric definitions of lower approximation, boundary and negative regions of a set. The main focus of the article is on the evaluation of the resulting set approximations and probabilistic decision tables using a number of proposed probabilistic measures. The application of the measures to evaluation of probabilistic decision tables is illustrated with a comprehensive example.

1 Introduction

In the rough set theory introduced by Pawlak, any undefinable or rough set, is approximately defined in terms of a pair of definable sets called lower and upper approximation respectively [1]. Any set is considered to be definable if it can be expressed as a union of equivalence classes (elementary sets) of an equivalence relation representing pre-existing ability to classify elements of the universe of interest into disjoint categories. The upper approximation is the least definable set containing the rough set. The lower approximation is the largest definable set contained in the rough set. In other words, any definable set has an exact description while any rough set has approximate description consisting of two exact descriptions of its lower and upper approximations.

The original Pawlak's formulations of lower and upper approximation are based on the set inclusion and set overlap operators respectively [1]. Although it is sufficient in many applications, there are situations in which the original, set inclusion operator-based formulation of lower approximation is too restrictive. To deal with this problem, the more relaxed formulation of lower approximation in terms of controlled degree of overlap between sets rather than the inclusion relation was introduced in [2] in the context of the variable precision rough set model (VPRS). In the VPRS formulation, the lower and upper approximations could be naturally interpreted in probabilistic terms, leading to generalized notions of rough set approximations, generalized concept of reduct, etc. (see, for instance, [2, 8]). In particular, probabilistic measures of quality and accuracy of approximation can be introduced to complement and generalize the original measures introduced by Pawlak [1].

In what follows, the basic definitions of the VPRS model are presented and compared to the original rough set model. For the purpose of comparison, the original rough set model is interpreted in probabilistic terms. This coincides with the recent results by Pawlak where probabilistic aspects of rough set model and of data-acquired decision tables are investigated in detail [3]. In particular, Pawlak demonstrates that basic Bayes' laws such as Total Probability Theorem and the Bayes' Theorem are preserved in decision tables computed from data.

These conclusions support the validity of the results presented in this paper since they rely on the Bayes' laws that are in practice applicable exclusively to empirical data represented in the form of probabilistic decision tables [4]. We note that the VPRS model leads to set approximations with broader positive region and narrower boundary region than the original model of rough sets. Based on the generalized set approximations, probabilistic measures of approximation quality are introduced and interpreted in the context of their potential applications to data analysis and predictive model derivation from data. Their application to analysis of probabilistic decision tables is illustrated with a comprehensive example.

2 Basic Notions of the VPRS model

In this section we provide basic definitions of the VPRS model and relate them to Pawlak's original model of rough sets. While doing that, we reformulate some basic definitions of the original model in probabilistic terms to clearly emphasize similarities and differences.

Let U denote a universe of objects referred to as elementary events and let $s(U)$ be the σ -algebra of measurable subsets of U referred to as random events. We will assume that there is a random process generating new objects e belonging to the universe U . For each new object e , we will say that the event $X \in s(U)$ occurred if the object $e \in X$. In addition, we will assume the existence of the *prior* probability function $P(X)$ on subsets X belonging to $s(U)$. We will also assume that all subsets under consideration in this article are members of the family of sets $s(U)$ and that all of them are likely to occur, that is that $P(X) > 0$, and that their occurrence is not certain, that is that $P(X) < 1$. These extra assumptions are justified by the fact that there is no need to construct a predictive model for events about which it is known that they are unlikely to occur or that they do occur with certainty.

To define the structure of rough approximation space, we will denote by R an equivalence relation on U with the finite number of equivalence classes (elementary sets) E_1, E_2, \dots, E_n such that $P(E_i) > 0$ for all $1 \leq i \leq n$. Clearly, the assumption of finite number of equivalence classes does not mean that the universe U is finite. Each elementary set E can be assigned a measure of overlap with the set X by the conditional probability function defined as $P(X|E) = \frac{P(X \cap E)}{P(E)}$. The values of the conditional probability function are normally estimated from sample data by taking the ratio $P(X|E) = \frac{\text{card}(X \cap E)}{\text{card}(E)}$. The *asymmetric* VPRS generalization of the original rough set model is based on the values of the probability function P and two lower and upper limit certainty threshold parameters l and u such that $0 \leq l < P(X) < u \leq 1$. Please note that the requirement $l < P(X)$ is an additional restriction on the values of the parameters which was not used in previous VPRS presentations. The justification behind this constraint will be explained after introduction of main VPRS definitions. The VPRS model is *symmetric* if $l = 1 - u$. Traditionally [2], in this special case the symbol β such that $0 < P(X) < \beta \leq 1$ is used instead of the symbol u to denote the model upper threshold parameter.

2.1 Positive Region

The parameter u defines the *u-positive region* or the *u-lower approximation* of the set X in the VPRS model. The value of u reflects the least acceptable degree of the conditional probability $P(X|E_i)$ to include the elementary set E_i in the positive region, or *u-lower approximation* of the set X . It can be perceived as quality threshold of the probabilistic information associated

with elementary sets E_i . Only those elementary sets with sufficiently high information quality are to be included in the positive region. The u -positive region of the set (event) X , $POS_u(X)$ is defined as

$$POS_u(X) = \cup\{E_i : P(X|E_i) \geq u\}.$$

Intuitively, u represents the desired level of improved prediction accuracy when predicting the occurrence of the event X based on the information that event E actually occurred. The lower approximation represents an area of the universe where the likelihood of X occurrence is *high*, where the subjective term *high* is reflected by the given value of the parameter u . Since the prior probability of X occurrence, that is in the absence of any additional information, is $P(X)$ then the improvement of prediction accuracy is possible only if $P(X) < u \leq 1$. This explains the constraint imposed on the value of u in the previous section. Clearly, other factors can be used to affect the value of u , such as the cost factors [5], but the general limitation $P(X) < u \leq 1$ has to be maintained independent of other factors.

One can also note that the lower approximation defined in the original Pawlak's model as $POS(X) = \cup\{E_i : X \supseteq E_i\}$ corresponds to $POS_1(X)$ of the VPRS model and that $POS(X) \subseteq POS_u(X)$ for all u such that $P(X) < u \leq 1$. This means that u -lower approximation is in a sense *broader* than the original rough set theory lower approximation of a set.

2.2 Negative Region

The l -negative region $NEG_l(X)$ of the set X in the VPRS approach, represents an area of the universe where the occurrence of the set X is significantly less likely than random guess (prior) probability $P(X)$. The threshold value for the probability of X occurrence is represented by the lower limit parameter l . The decrease in the probability of X occurrence relative to prior probability is possible only if $0 \leq l < P(X)$ that justifies the constraint imposed on l in our initial assumptions. Formally,

$$NEG_l(X) = \cup\{E_i : P(X|E_i) \leq l\}.$$

Alternatively, the l -negative region can be perceived as $(1 - l)$ -positive region of the complement of the set X , the set $\neg X = U - X$. This is due to the fact that

$$\begin{aligned} NEG_l(X) &= \cup\{E_i : P(X|E_i) \leq l\} = \cup\{E_i : 1 - P(X|E_i) \geq 1 - l\} = \\ &= \cup\{E_i : P(\neg X|E_i) \geq 1 - l\} = POS_{1-l}(\neg X). \end{aligned}$$

Consequently, $NEG_l(X)$ includes all those elementary sets E_i that are associated with sufficiently high probability to predict that X would not occur.

The original Pawlak's notion of negative region defined as

$$NEG_{ORG}(X) = \cup\{E_i : X \cap E_i = \emptyset\}$$

can be reformulated in probabilistic terms as

$$NEG_{ORG}(X) = \cup\{E_i : P(X|E_i) = 0\} = NEG_0(X).$$

Clearly, $NEG_0(X) \subseteq NEG_l(X)$ for all $0 \leq l < P(X)$ which means that in general the l -negative region is broader than the original rough set model negative region.

2.3 Boundary Region

The boundary region $BND_{l,u}(X)$ in the VPRS model covers an area of the universe where the prediction about the occurrence of the event (set) X or its complement $\neg X$ is not possible

with sufficient likelihood to qualify either for the positive region of X or the positive region of the complement $\neg X$ (that is the negative region of X). This can be expressed formally as

$$BND_{l,u}(X) = \cup\{E_i : u > P(X|E_i) > l\}, \text{ or equivalently as}$$

$$BND_{l,u}(X) = \cup\{E_i : P(X|E_i) < u \wedge P(\neg X|E_i) < 1 - l\}$$

The boundary region of the original rough set model can be defined using the same notation by

$$BND_{ORG}(X) = \cup\{E_i : 1 > P(X|E_i) > 0\} = BND_{0,1}(X).$$

We can notice that the VPRS boundary area is in general *narrower* than the original rough set boundary area since for all permissible l, u values and sets X we have

$$BND_{l,u}(X) \subseteq BND_{ORG}(X).$$

2.4 Upper Approximation

The *l-upper approximation* $UPP_{ORG}(X)$ of the set X in the original rough set model isolates an area of the universe where the occurrence of the target set (event) X is possible, that is, it could be predicted with any probability greater than zero. This is formulated in the definition

$$UPP_{ORG}(X) = \cup\{E_i : X \cap E_i \neq \emptyset\}$$

which in probabilistic terms can be re-expressed as

$$UPP_{ORG}(X) = \cup\{E_i : P(X|E_i) > 0\} = POS_{ORG}(X) \cup BND_{ORG}(X).$$

The natural generalization of that definition in the framework of VPRS is to define the upper approximation as a union of VPRS positive and boundary regions. This leads to the following generalized definition of *l-upper approximation*:

$$UPP_l(X) = \cup\{E_i : P(X|E_i) > l\} = POS_u(X) \cup BND_{l,u}(X).$$

We may notice that in the original rough set model we always have $P(UPP(X)) > 0$. This is also the case in the VPRS model as demonstrated by the following Proposition.

Proposition 1.

For any $0 \leq l < P(X)$ the following property $P(UPP_l(X)) > 0$ is always true.

Proof:

Assume that $P(UPP_l(X)) = 0$ for a certain specific $l = a$ and $u = b$. This means that both $POS_b(X) = \emptyset$ and $BND_{a,b}(X) = \emptyset$. Consequently, all elementary sets E_i must belong to the negative region $NEG_a(X)$. That conclusion however leads to a contradiction since it would mean that for all $1 \leq i \leq n$ we would have $\frac{P(X \cap E_i)}{P(E_i)} \leq l$, that is $P(X \cap E_i) \leq lP(E_i)$. Since $\sum_i P(X \cap E_i) = P(X)$ and $\sum_i P(E_i) = 1$ we get $P(X) \leq l$ that contradicts our assumption.

To conclude this section, we would like to emphasize that definitions of VPRS model rough regions are derived based solely on the assessment of the quality of probabilistic information associated with elementary sets of the universe U . The basic definitions of the original model of rough sets can also be interpreted in that way. However, the information quality measures are different in the original rough set model as discussed in the next section.

3 Model Quality Measures

In practical applications of the VPRS model to data mining, machine learning and data-based modelling problems, it is essential to evaluate the quality of the available information and derived models using appropriate measures. The measures are intended to capture essential characteristics of the universe U and of the target set (event) X with respect to their suitability

for building useful classifiers and to acquire better understanding of relations occurring in the universe. Some initial measures such as *accuracy* or *roughness* were introduced by Pawlak [1] and later expanded by others in the framework of the original rough set model (see, for instance, [9]). In the VPRS approach, all measures are probabilistic and in practice computed from representative sample data, as it is common in statistical analysis. The measures can be divided into two categories: local measures associated with individual elementary sets and global measures operating on groups of elementary sets, for example contained in the lower approximation of the target set. In what follows, we introduce the measures and relate them, whenever possible, to equivalent measures used in the original rough set theory.

3.1 Local Measures

The local measures are associated with elementary sets E_1, E_2, \dots, E_n partitioning the universe U . All these measures are relative with respect to the target set (event) $X \subseteq U$.

3.1.1 Conditional Probability

The single most important measure is the *conditional probability* of the event X given by

$$P(X|E) = \frac{P(X \cap E)}{P(E)}.$$

This measure was introduced in the context of rough set theory in [6]. It is also known as *rough membership function*. All other measures presented here are derivatives of conditional probability measure. The conditional probability measure is the basic chunk of information used in all decision making in this approach. It can be interpreted as a measure of certainty of predictions, when predicting X with this probabilistic information.

3.1.2 Certainty Gain

To evaluate the degree of increase (or decrease) of the predictive probability relative to the random guess (prior) probability $P(X)$, based on the conditional probability information associated with an elementary set E , the *certainty gain* measure $G(X|E)$ is introduced as

$$G(X|E) = \frac{P(X|E) - P(X)}{P(X)} = \frac{P(X|E)}{P(X)} - 1 \text{ if } P(X|E) > P(X) \text{ and}$$

$$G(X|E) = \frac{P(X) - P(X|E)}{1 - P(X)} = \frac{1 - P(X|E)}{1 - P(X)} - 1 \text{ if } P(X|E) < P(X).$$

One may notice that the certainty gain value in the original rough set model is given by

$$G(X|E) = \frac{1}{P(X)} - 1 > 0 \text{ for all elementary sets } E \text{ belonging to the positive region and}$$

$$G(X|E) = \frac{1}{1 - P(X)} - 1 > 0 \text{ for all elementary sets } E \text{ belonging to the negative region of the set } X.$$

In the VPRS approach, the certainty gain value is positive in both the positive and negative regions. In addition, it satisfies the following relations:

$G(X|E) \geq \frac{u}{P(X)} - 1 > 0$ where $P(X) < u \leq 1$ for all elementary sets E belonging to the lower approximation of the set X , and

$G(X|E) \geq \frac{1-l}{1-P(X)} - 1 > 0$ where $0 \leq l < P(X)$ for all elementary sets E belonging to the positive region of the set X .

3.2 Global Measures

The global measures capture aggregate distribution, coverage and gain information about rough approximation regions and rough approximations to have clearer idea about the quality of available data and potential data-derived models. In what follows, we introduce the basic measures and relate them to already known measures used in the rough set theory, whenever applicable.

3.2.1 Probabilities of Approximation Regions

The probability $P(UPP_l(X))$ of upper approximation $UPP_l(X)$ of the set X represents the relative size of this region. It was earlier introduced in the framework of the original rough set theory as *coverage*. It can be computed from probabilities of elementary sets included in $UPP_l(X)$ by

$$P(UPP_l(X)) = \sum_{E \subseteq UPP_l(X)} P(E).$$

Similarly, the probabilities of u -positive, (l, u) -boundary and l -negative regions can be computed respectively by

$$P(POS_u(X)) = \sum_{E \subseteq POS_u(X)} P(E) \text{ for positive region;}$$

$$P(BND_{l,u}(X)) = \sum_{E \subseteq BND_{l,u}(X)} P(E) \text{ for boundary region and}$$

$$P(NEG_l(X)) = \sum_{E \subseteq NEG_l(X)} P(E) \text{ for negative region.}$$

The sum $\gamma_{l,u}(X) = P(POS_u(X)) + P(NEG_l(X))$ is a dependency measure reflecting the relative size of the positive and negative region in the universe. It can be used as a measure of accuracy of approximate representation of the set X . This measure is equivalent to $P(BND_{l,u}(X))$ since $P(BND_{l,u}(X)) = 1 - \gamma_{l,u}(X)$.

Knowing the probabilities of rough regions helps in understanding the distribution of the regions, approximation quality and the relative positioning of the set X within the universe U .

3.2.2 Accuracy and Roughness of Approximation

As in the Pawlak's definition [1], the *accuracy* $ACC_{l,u}(X)$ of set X approximation can be expressed as a ratio of "sizes" of lower to upper approximations of the set. In the VPRS approach, the cardinality of a set is replaced with the probability measure leading to the following definition of accuracy function:

$$ACC_{l,u}(X) = \frac{P(POS_u(X))}{P(UPP_l(X))}.$$

Because $POS_u(X) \subseteq UPP_l(X)$, the accuracy of approximation is the same as the conditional probability $P(POS_u(X)|UPP_l(X))$ of positive region $POS_u(X)$ given that the upper approximation event $UPP_l(X)$ occurred.

An alternative measure is *roughness*, defined after Pawlak as $\rho_{l,u} = 1 - ACC_{l,u}(X)$. The roughness represents the degree of uncertainty in set X approximate representation. In probabilistic terms, the roughness can be interpreted as the conditional probability

$P(BND_{l,u}(X)|UPP_l(X))$ of boundary region $BND_{l,u}(X)$ given that the upper approximation $UPP_l(X)$ occurred. In the practice, it is desirable to minimize roughness while maximizing the accuracy of approximation.

3.2.3 Coverage Measures

In the original rough set model the *precision* measure is defined as the ratio of the size of the lower approximation of the target set X to the size of the set X [1]. In the VPRS model, lower approximation of a set X is not necessarily a subset of the set X . Consequently, the natural generalization of the original precision measure is the *coverage of lower approximation* defined as conditional probability $P(POS_u(X)|X)$ of positive region of the set X given the occurrence of the set. The coverage of lower approximation can be simply computed by

$$COV_{POS_u}(X) = P(POS_u(X)|X) = \frac{P(POS_u(X) \cap X)}{P(X)}, \text{ that is}$$

$$COV_{POS_u}(X) = \frac{1}{P(X)} \sum_{E \subseteq POS_u(X)} P(E)P(X|E).$$

Similarly, the coverage of negative region can be computed by

$$COV_{NEG_l}(X) = P(NEG_l(X)|X) = \frac{P(NEG_l(X) \cap X)}{P(X)}, \text{ that is}$$

$$COV_{NEG_l}(X) = \frac{1}{P(X)} \sum_{E \subseteq NEG_l(X)} P(E)P(X|E).$$

The above measure provides an estimate of frequency of occurrences of new objects (observations) belonging to positive region or negative regions in situations when they do belong to the set X . The value of the coverage of the positive region should be maximized because high value means that high percentage of objects (observations) will likely be properly classified as likely members of the set X , based on the easily verifiable fact that they do belong to the positive region $POS_u(X)$. On the other hand, the coverage of the negative region should be minimized since lower values mean that lower percentage of objects belonging to X will be improperly classified as likely not belonging to X .

In principle, the coverage measure can be computed with respect to any discernible region of the universe. For instance, the *coverage of boundary region*, given by

$$COV_{BND_{l,u}}(X) = P(BND_{l,u}(X)|X) = \frac{P(BND_{l,u}(X) \cap X)}{P(X)}, \text{ that is}$$

$$COV_{BND_{l,u}}(X) = \frac{1}{P(X)} \sum_{E \subseteq BND_{l,u}(X)} P(E)P(X|E)$$

will tell us what percentage of cases belonging to X will fall into the boundary $BND_{l,u}(X)$.

Alternatively, this will be the percentage of X cases about which we will be able to predict X with the probability higher than l but lesser than u . In practical applications, the boundary area should be minimized. Similarly, the *coverage of upper approximation* can be defined as conditional probability $P(UPP_l(X)|X)$ of l -upper approximation, given the occurrence of the set X .

3.2.4 Certainty Measures

The expected certainties $CERT_{region}(X)$ of prediction of the event X are average measures extending over upper approximation and positive and boundary regions. They provide general information about expected success rate when predicting that an object belongs to X , based on an observation that it belongs to one of the approximation regions. The measures can be computed using probabilities of elementary sets included in the respective regions.

The *certainty of positive region* provides aggregate information about the odds of an object belonging to the set X if it is known that it belongs to lower approximation of X . It is defined by

$$CERT_{POS_u}(X) = P(X|POS_u(X)) = \frac{P(POS_u(X) \cap X)}{P(POS_u(X))}, \text{ that is}$$

$$CERT_{POS_u}(X) = \frac{1}{P(POS_u(X))} \sum_{E \subseteq POS_u(X)} P(E)P(X|E).$$

This measure can be used as an estimator of the expected performance of a prediction system for predictions involving cases belonging to the lower approximation $POS_u(X)$.

The *certainty of negative region* is defined as

$$CERT_{NEG_l}(X) = P(X|NEG_l(X)) = \frac{P(NEG_l(X) \cap X)}{P(NEG_l(X))}, \text{ that is}$$

$$CERT_{NEG_l}(X) = \frac{1}{P(NEG_l(X))} \sum_{E \subseteq NEG_l(X)} P(E)P(X|E).$$

In practical applications with probabilistic decision tables [4], the certainty of positive region should be maximized whereas the certainty of negative region should be minimized.

The *certainty of upper approximation* represents general information about expected success rate of predictions based on the upper approximation:

$$CERT_{UPP_l}(X) = P(X|UPP_l(X)) = \frac{P(UPP_l(X) \cap X)}{P(UPP_l(X))}, \text{ or equivalently}$$

$$CERT_{UPP_l}(X) = \frac{1}{P(UPP_l(X))} \sum_{E \subseteq UPP_l(X)} P(E)P(X|E).$$

Similarly, the *certainty of boundary region* is a measure of the average success rate when making predictions based on information represented by elementary sets of the region.

$$CERT_{BND_{l,u}}(X) = P(X|BND_{l,u}(X)) = \frac{P(BND_{l,u}(X) \cap X)}{P(BND_{l,u}(X))}, \text{ that is}$$

$$CERT_{BND_{l,u}}(X) = \frac{1}{P(BND_{l,u}(X))} \sum_{E \subseteq BND_{l,u}(X)} P(E)P(X|E).$$

Finally, we may notice that the coverage and certainty measures are related to each other by the following formulas:

$$CERT_{POS_u}(X) = \frac{P(X)}{P(POS_u(X))} COV_{POS_u}(X),$$

$$CERT_{NEG_l}(X) = \frac{P(X)}{P(NEG_l(X))} COV_{NEG_l}(X),$$

$$CERT_{UPP_l}(X) = \frac{P(X)}{P(UPP_l(X))} COV_{UPP_l}(X) \text{ and}$$

$$CERT_{BND_{l,u}}(X) = \frac{P(X)}{P(BND_{l,u}(X))} COV_{BND_{l,u}}(X).$$

The above formulas indicate that these two measures are essentially equivalent although their interpretation is different.

3.2.5 Certainty Gain Measure

Similarly to other global measures, one can define expected certainty gain $G_{region}(X)$ measure for approximation regions. This measure would capture the average degree of improvement in prediction probability when using the region's definition in *probabilistic decision table* [4] (or, equivalently, in *decision algorithm* [3]). For the positive region, it is defined as follows:

$$G_{POS_u}(X) = \sum_{E \subseteq POS_u(X)} P(E|POS_u(X))G(X|E), \text{ or in equivalent form}$$

$$G_{POS_u}(X) = \frac{1}{P(POS_u(X))} \sum_{E \subseteq POS_u(X)} P(E)G(X|E).$$

For the negative region the certainty gain is

$$G_{NEG_l}(X) = \sum_{E \subseteq NEG_l(X)} P(E|NEG_l(X))G(X|E), \text{ or in equivalent form}$$

$$G_{NEG_l}(X) = \frac{1}{P(NEG_l(X))} \sum_{E \subseteq NEG_l(X)} P(E)G(X|E).$$

Interestingly, the expected gain measure is essentially equivalent to the coverage and certainty measures for the positive and negative regions as defined in subsections 3.2.3 and 3.2.4.

It can be shown that

$$G_{POS_u}(X) = \frac{1}{P(X)} CERT_{POS_u}(X) - 1,$$

$$G_{POS_u}(X) = \frac{1}{P(POS_u(X))} COV_{POS_u}(X) - 1,$$

$$G_{NEG_l}(X) = \frac{P(X)}{1-P(X)} (1 - \frac{1}{P(X)} CERT_{NEG_l}(X)),$$

$$G_{NEG_l}(X) = \frac{P(X)}{1-P(X)} (1 - \frac{1}{P(NEG_l(X))} COV_{NEG_l}(X)),$$

which means that one measure can substituted for the other if probabilities $P(X)$, $P(POS_u(X))$ and $P(NEG_l(X))$ are known. To evaluate the expected certainty gain associated with the whole decision table, the measure of *global certainty gain* $G(X)$ can be used:

E_i	A_1	A_2	A_3	A_4	$P(X E_i)$	$P(E_i)$	$G(X E_i)$	R-REGION
E_1	3	0	1	3	0.0	0.15	1.1044	NEG
E_2	2	1	0	3	0.75	0.05	0.4291	BND
E_3	3	1	0	3	0.0	0.10	1.1044	NEG
E_4	1	1	1	1	1.0	0.01	0.9054	POS
E_5	2	1	1	2	0.82	0.24	0.5625	POS
E_6	2	0	0	3	0.12	0.05	0.8518	BND
E_7	2	0	1	1	0.92	0.15	0.7530	POS
E_8	3	0	0	1	0.91	0.15	0.7340	POS

Table 1: Probabilistic Decision Table with $P(X) = 0.5248$, $l = 0.8$ and $u = 0.8$

$$G(X) = \sum_{E \subseteq U} P(E)G(X|E).$$

When deriving decision table from data, one of the primary guiding objectives should be maximizing the global certainty gain of the table. The global certainty gain is maximized if the target X is a precise or definable set[1]. In this case, the maximum value of the global certainty gain $G_{\max}(X)$ is given by:

$$G_{\max}(X) = P(POS^*(X)) \frac{1-P(X)}{P(X)} + P(NEG^*(X)) \frac{P(X)}{1-P(X)} \text{ where}$$

$$POS^*(X) = \cup \{E_i : P(X|E_i) > P(X)\} \text{ and}$$

$$NEG^*(X) = \cup \{E_i : P(X|E_i) < P(X)\}.$$

The ratio of the actual certainty gain to the maximum certainty gain achievable defines the *relative global certainty gain* $G_{rel}(X)$:

$$G_{rel}(X) = \frac{G(X)}{G_{\max}(X)}.$$

The relative certainty gain provides normalized value of the global certainty gain which gives an idea about the overall degree of predictive certainty improvement obtained with a decision table.

4 Evaluation of Probabilistic Decision Tables

The *probabilistic decision table* is a tabular representation of the available information about the universe U and about the relation existing between the classes E_1, E_2, \dots, E_n of the approximation space and the target set X . The information is typically acquired from sample data but it is also possible to construct the decision table using other sources of knowledge, for example accumulated experience of a domain expert. In the probabilistic decision table it is assumed that objects are represented by features (attributes) taking finite number of values. The objects are grouped into classes E_1, E_2, \dots, E_n based on the identity of attribute values. The decision table summarizes the following information for each elementary set E_i :

- the unique identification of the class E_i ;
- the combination of attribute values common to all objects in the class;
- the estimate of the conditional probability $P(X|E_i)$ relative to the target set X ;
- the estimate of the probability $P(E_i)$ of the class E_i ;
- the value of the certainty gain measure $G(X|E_i)$;
- the specification of the rough approximation region (*R-Region*) the class E_i is included in, based on the given values of the precision control parameters l and u .

In the following example, the Table 1 is a probabilistic decision table that was obtained assuming that $l = 0.1$ and $u = 0.8$ with $P(X) = 0.5248$.

The global measures introduced in previous sections can be used to evaluate probabilistic decision tables. In fact, in the typical absence of complete and precise information about the universe U , the evaluation measures applied to the decision tables obtained from sample data provide an estimate of the actual values of the measures as they exist in the universe U . The probabilities appearing in the decision tables can be estimated in a standard way using the ratios of cardinalities of appropriate data sets.

Prior to calculating global measures from the decision table given in Table 1, we can summarize the approximation regions of the target set X and their respective probabilities:

- upper approximation $UPP_{0.1}(X) = E_2 \cup E_4 \cup E_5 \cup E_6 \cup E_7 \cup E_8$ with $P(UPP_{0.1}(X)) = 0.75$;
- 0.8-positive region with $POS_{0.8}(X) = E_4 \cup E_5 \cup E_7 \cup E_8$ and $P(POS_{0.8}(X)) = 0.55$;
- (0.1, 0.8)-boundary region $BND_{0.1,0.8}(X) = E_2 \cup E_6$ with $P(BND_{0.1,0.8}(X)) = 0.1$;
- 0.1-negative region $NEG_{0.1}(X) = E_1 \cup E_3$ with $P(NEG_{0.1}(X)) = 0.25$.

From the above figures and the information given in Table 1, we can calculate global quality measures as follows:

- the accuracy of set X approximation:

$$ACC_{0.1,0.8}(X) = \frac{P(POS_{0.1,0.8}(X))}{P(UPP_{0.1}(X))} = \frac{0.55}{0.75} = 0.733;$$

- the coverage of positive region:

$$COV_{POS_{0.8}}(X) = \frac{1}{P(X)} \sum_{E \subseteq POS_{0.8}(X)} P(E)P(X|E) = \frac{0.4813}{0.5284} = 0.911;$$

- the coverage of boundary region:

$$COV_{BND_{0.1,0.8}}(X) = \frac{1}{P(X)} \sum_{E \subseteq BND_{0.1,0.8}(X)} P(E)P(X|E) = \frac{0.0435}{0.5284} = 0.082;$$

- the coverage of negative region:

$$COV_{NEG_{0.1}}(X) = \frac{1}{P(X)} \sum_{E \subseteq NEG_{0.1}} P(E)P(X|E) = 0;$$

- the certainty of positive region:

$$CERT_{POS_{0.8}}(X) = \frac{1}{P(POS_{0.8}(X))} \sum_{E \subseteq POS_{0.8}(X)} P(E)P(X|E) = \frac{0.4813}{0.55} = 0.875;$$

- the certainty of upper approximation:

$$CERT_{UPP_{0.1}}(X) = \frac{P(X)}{P(UPP_{0.1}(X))} = \frac{0.5284}{0.75} = 0.699;$$

- the certainty of boundary region:

$$CERT_{BND_{0.1,0.8}}(X) = \frac{1}{P(BND_{0.1,0.8}(X))} \sum_{E \subseteq BND_{0.1,0.8}(X)} P(E)P(X|E) = \frac{0.0435}{0.1} = 0.435;$$

- the certainty of negative region:

$$CERT_{NEG_{0.1}}(X) = \frac{P(X)}{P(NEG_{0.1}(X))} COV_{NEG_{0.1}}(X) = 0;$$

- the expected certainty gain of the positive region:

$$G_{POS_{0.8}}(X) = \frac{1}{P(POS_{0.8}(X))} COV_{POS_{0.8}}(X) - 1 = \frac{1}{0.55} 0.911 - 1 = 0.656;$$

- the expected certainty gain of the negative region:

$$G_{NEG_{0.1}}(X) = \frac{P(X)}{1-P(X)} (1 - \frac{1}{P(NEG_{0.1}(X))} COV_{NEG_{0.1}}(X)) = \frac{0.5248}{0.4752} = 1.1044;$$

- the global certainty gain:

$$G(X) = \sum_{E \subseteq U} P(E)G(X|E) = 0.7071;$$

- the maximum achievable certainty gain:

$$G_{\max}(X) = P(POS^*(X)) \frac{1-P(X)}{P(X)} + P(NEG^*(X)) \frac{P(X)}{1-P(X)} = 0.8843;$$

- the normalized global certainty gain:

$$G_{rel}(X) = \frac{G(X)}{G_{\max}(X)} = 0.7996.$$

The results of the analysis of the example decision table indicate that it has rather favorable characteristics such as good accuracy, high coverage of the positive region, low coverage of the boundary and negative regions, relatively high average certainty and certainty gain in the positive and negative regions, high global certainty gain and low probability of the boundary region.

5 Final Remarks

The article attempts to define and summarize all measures applicable to analysis of approximation spaces, rough sets and probabilistic decision tables author believes are essential for their proper evaluation. All measures are defined using probability theory notation as a result of using VPRS model of rough sets. Some of the measures presented here, such as for instance certainty gain measure, to the author's best knowledge, have never been published before. Some other measures, such as coverage measures, are generalizations of measures defined before. The need to define and use number of different measures for evaluation of decision tables arose as a result of author's experiences with numerous practical problems involving using data to create decision tables. The measures summarized in this paper seem to make it easier to make distinction between good versus not very useful decision tables.

6 Acknowledgments

The research reported in this article was supported in part by a research grant awarded by Natural Sciences and Engineering Council of Canada. Many thanks to Zdzisław Pawlak, Arul Siromoney and Dominik Slezak for valuable comments and suggestions.

References

- [1] Pawlak, Z. (1991) *Rough Sets: Theoretical Aspects of Reasoning About Data*. Kluwer Academic Publishers.
- [2] Ziarko, W. (1993) *Variable Precision Rough Sets Model*. Journal of Computer and Systems Sciences, vol. 46, no. 1, pp. 39-59.
- [3] Pawlak, Z. (2002) *Rough sets and decision algorithms*. In: Ziarko, W. and Y. Yao (eds). *Rough Sets and Current Trends in Computing*, Lecture Notes in AI 2005, Springer Verlag 2001, 30-45.
- [4] Ziarko, W. *Probabilistic decision tables in the variable precision rough set model*. Computational Intelligence: an International Journal, vol. 17, no 3, 2001, 593-603.
- [5] Ziarko, W. *Decision making with probabilistic decision tables*. In: Zhong, N. Skowron, A. Ohsuga, S. (eds.) *New Directions in Rough Sets, Data Mining, and Granular-Soft Computing*, Lecture Notes in AI 1711, Springer Verlag, 463-471.
- [6] Wong, S.K.M. Ziarko, W. *Comparison of the probabilistic approximate classification and the fuzzy set model*. International Journal for Fuzzy Sets and Systems, vol. 21, 1986, 357-362.
- [7] Wong, S.K.M. Ziarko, W. *On learning and evaluation of decision rules in the context of rough sets*. Proceedings of the International Symposium on Methodologies for Intelligent Systems, Knoxville, 1986, 308-224.
- [8] Beynon, M. (2002) *An Investigation of β reduct selection within the variable precision rough set model*. In: Ziarko, W. and Yao, Y. (eds). *Rough Sets and Current Trends in Computing*, Lecture Notes in AI 2005, Springer Verlag 2001, 30-45.
- [9] Tsumoto, S. (1998) *Modelling medical diagnostic rules based on rough sets*. In: Polkowski, L. Skowron, A. (eds.) *Rough Sets and Current Trends in Computing*, Lecture Notes in AI 1424, Springer Verlag 2001, 475-482.

A Data Preparation Bayesian Approach for a Clustering Genetic Algorithm

Estevam R. HRUSCHKA Jr.

*COPPE / Universidade Federal do Rio de Janeiro, Caixa Postal 68.506, CEP 21.945-970,
Rio de Janeiro, RJ, Brasil.*

Eduardo R. HRUSCHKA

*Universidade Tuiuti do Paraná, Av. Comendador Franco, 1.860, CEP 80.215-909,
Curitiba, PR, Brasil.*

Nelson F. F. EBECKEN

*COPPE / Universidade Federal do Rio de Janeiro, Caixa Postal 68.506, CEP 21.945-970,
Rio de Janeiro, RJ, Brasil.*

Abstract. The substitution of missing values is an important task in data mining applications and it can be performed by means of many methods. This work describes the use of the bayesian algorithm K2 as a data preparation tool for a clustering genetic algorithm. We illustrate the proposed method by means of simulations in three datasets: Ruspini, 200 Randomly Generated and Wisconsin Breast Cancer. The obtained results show that the substitution Bayesian method is adequate in the Clustering Genetic Algorithm context.

1. Introduction

Knowledge discovery in databases is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data [1]. In this context, data mining is a step in this process that centers on the automated discovery of new facts and relationships in data and it consists of three basic steps: data preparation, information discovery and analysis of the mining algorithm output [2].

The substitution of missing values is an important task in the data preparation step and it can be performed by means of many methods. The most common methods used to this important task are regression, bayesian inference and decision trees [3]. This work uses the bayesian algorithm K2 to substitute missing values. Basically, a Bayesian network may be seen as a classification/regression model [4] and the substitution of missing values may be seen as a prediction task [5]. Thus, when having prior information about the data (i.e. dataset), Bayesian networks can be used to predict the most suitable values to substitute the missing ones. Besides, Bayesian networks are usually employed in data mining learning tasks mainly because they [6]: (i) may deal with incomplete data sets in a direct way; (ii) can learn causal relationships; (iii) may combine *a priori* knowledge with patterns learned from data and (iv) help to avoid overfitting. However, the knowledge acquired by such models is usually not so comprehensible for humans as other possible structures like classification rules [7]. This fact can be a major obstacle in domains such

as data mining where it is important to have symbolic rules or other forms of knowledge structure [8].

The Clustering Genetic Algorithm (CGA) [9] was developed to the knowledge discovery process and it can provide “If...Then” propositional rules. Since these rules are the simplest and most comprehensible, they should be tried first [10]. Therefore, we believe that the combination of a data preparation Bayesian approach with the CGA can provide a useful hybrid method for data mining applications. This work shows the soundness of the substitution proposed method in three datasets: Ruspini, 200 Randomly Generated and Wisconsin Breast Cancer.

The next section describes how the data preparation task is performed and section 3 describes the CGA. Section 4 shows the simulation results on three datasets, whereas the section 5 describes the conclusions and points out some future works.

2. Data Preparation – missing values substitution process

There are many techniques to deal with missing values [5]. In this work, we employ a substitution method that is similar to the one described in [11]. Basically, the main difference is that we use a Bayesian network instead of a decision tree. Thus, a Bayesian network is constructed in order to infer the most suitable values to fill in the sample gaps produced by the missing values.

The substitution process starts with the selection (from the original sample) of all the sample objects that do not have missing data. These objects form a *clean sample* that is used as a *clean training dataset* to the Bayesian network construction. Afterwards, all the features with missing values are identified and for each one - now considered as a target feature - a Bayesian network is generated. It means that if we have 3 features with missing values, three *training clean samples* are generated. Subsequently, the *K2* algorithm [12] is applied to construct the Bayesian networks, which are used to substitute the missing values. Thus, the substitution process can be summarized as follows:

1. Identify the features that present missing values;
2. Generate *clean training* datasets for each identified feature;
3. Construct a Bayesian network with each *clean training* dataset;
4. Infer the best values to substitute the missing ones in the original sample.

3. Clustering Genetic Algorithm

Clustering is a task where one seeks to identify a finite set of categories or clusters to describe the data. This work considers that clustering involves the partitioning of a set X of objects into a collection of mutually disjoint subsets C_i of X . Formally, let us consider a set of N objects $X = \{X_1, X_2, \dots, X_N\}$ to be clustered, where each $X_i \in \mathcal{R}^p$ is an attribute vector consisting of “ p ” real measurements. The objects must be clustered into non-overlapping groups $C = \{C_1, C_2, \dots, C_k\}$ where k is the number of clusters, such that:

$$C_1 \cup C_2 \cup \dots \cup C_k = X, \quad C_i \neq \emptyset, \quad \text{and} \quad C_i \cap C_j = \emptyset \quad \text{for } i \neq j. \quad (1)$$

The problem with finding an optimal solution to the partition of N data into C classes is NP-complete [13] and because the number of distinct partitions of n objects into m clusters increases approximately as $m^n/m!$, attempting to find a globally optimum solution is usually not computationally feasible [14]. Genetic algorithms are widely believed to be effective on NP-complete global optimization problems and they can provide good sub-optimal solutions in reasonable time [15]. Thus, a clustering genetic algorithm can

provide a way of finding the right clustering. Some approaches that describe the application of genetic algorithms to clustering problems can be found in [13,15,16].

There are several complications concerning the application of traditional genetic algorithms for clustering problems. The use of a simple encoding scheme, which yields to constant-length chromosomes, usually causes the problems of redundant codification and context insensitivity [13]. However, the CGA uses such schemes and avoids all of their problems [9].

3.1. Encoding Scheme

Considering that there are N objects to be clustered, a phenotype is represented as a one dimensional integer array with $(N+1)$ elements. As each data unit can be numbered from 1 to N , the i th element of a genotype represents the i th data unit, whereas the last gene represents the number of clusters. Therefore, each gene of a chromosome has a value over the alphabet $\{1,2,3,\dots,k\}$, where k is the maximum number of clusters. For example, considering a dataset with 20 objects one can get the following clustering: 22345123453321454552 5.

This means that the cluster whose label is 2 is formed by 5 objects $\{1,2,7,13,20\}$, and the cluster whose label is 1 has 2 objects $\{6,14\}$. The last gene corresponds to the number of clusters encoded by the solution. The first problem with this straightforward encoding scheme is that it is redundant (there are $5!$ chromosomes encoding the same solution to the original problem). Thus, the search space is much larger than the original one. In addition, there is another problem caused by this encoding scheme when it is applied with the traditional genetic operators. It is the undesirable effect of casting context-dependent information *out of context* under the standard crossover, i.e. the context insensitivity [13]. This problem happens when one is dealing with the classic genetic operators of crossover and mutation, which usually produce invalid solutions. To avoid these problems, we have developed genetic operators that are group-oriented. Basically, the developed CGA is as described in Figure 1.

- 1) Initialize a population of genotypes;
- 2) Evaluate each genotype in the population;
- 3) Apply a linear normalization;
- 4) Select genotypes by proportional selection;
- 5) Apply crossover and mutation;
- 6) Replace the old genotypes by the ones formed by 5);
- 7) If the convergence is attained, stop; if not, go to step 2).

Figure 1. Clustering Genetic Algorithm (CGA).

3.2. Objective Function

Determining the optimal number of clusters in a dataset is one of the most difficult aspects in the clustering process [16]. Most of the approaches found in the literature determine the right number of clusters according to criteria based on given clusterings. However, the CGA is suppose to optimize not only the clusterings for a given number of clusters but also the number of clusters. Therefore, the CGA employs an objective function based on the Average Silhouette Width [17]. Basically, let us consider an object "i" belonging to cluster A. So the average dissimilarity of "i" to all other objects of A is denoted by $a(i)$. Now consider a different cluster C and let us calculate the average dissimilarity of "i" to all objects of C, which will be here denoted by $d(i,C)$. After computing the $d(i,C)$ for all clusters $C \neq A$ we select the smallest of those, $b(i) = \min d(i,C)$ for $C \neq A$. This number represents the dissimilarity of "i" to its neighbor cluster. Now one defines the silhouette $s(i)$ like follows:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad (2)$$

When the cluster A contains only one object we consider that $s(i)=0$, which is the most neutral choice [17]. In addition, it is easy to see that: $-1 \leq s(i) \leq 1$. The objective function is the average of $s(i)$ for $i=1,2,\dots,N$. It implies that the best value of “k” happens when the objective function value is as high as possible.

3.3. Selection

The genotypes that make part of each generation are selected according to the roulette wheel selection strategy. As this strategy does not allow negative objective function values, a constant number equals to one is summed up to each objective function value before the selection process takes place. Besides, the best genotype is always copied into the succeeding generation.

3.4. Operators

Classic genetic operators are not suitable for clustering problems [13] because they usually work with a set of unrelated genes. In clustering problems the genes are related, i.e. equal allele genes mean that the objects they represent belong to the same cluster. In other words, the groups are the meaningful building blocks of a solution. Thus, it is necessary to use operators that work with groups of genes, instead of with the genes themselves. This fact leads us to develop operators referred to as group oriented [9].

The crossover operator combines together pieces of information coming from different genotypes and it works in the following way: First, two genotypes (A and B) are selected; Second, considering that A represents k_1 clusters, the CGA chooses randomly n $[1, k_1]$ clusters to copy in B. The unchanged groups of B are maintained and the changed ones have their objects allocated to the cluster that has the nearest centroid. In this way the child C is obtained. This same process is employed to get child D, but now considering that the changed clusters of B are copied in A.

There are two operators for mutation. The operator 1, which only works in clusterings formed by more than two groups, eliminates a randomly chosen group and places all their objects to the remaining cluster that has the nearest centroid. The operator 2 divides a randomly selected group into two new ones. The first group is formed by the objects closer to the centroid, whereas the other group is formed by those objects nearer to the farthest object to the centroid. These operators are suitable for grouping problems because they just change the genotypes in the smallest possible way, i.e. dividing groups and eliminating others in the hope of finding better solutions to the problem being solved. Besides, 50% of the genotypes are crossed-over, 25% are mutated by operator 1 and 25% are mutated by operator 2.

3.5. Initial Population

Fundamentally, we have also employed the methodology developed in [17] to set up the initial population. The initial clustering process is based on the selection of representative objects. The first selected object is the most centrally located in the set of objects. Subsequently, other objects are selected. Basically, the chance of selecting an object increases when it is far from the previously selected ones and when there are many objects next to it. After selecting the representative objects, the initial population is formed considering that the nonselected objects must be clustered according to their

proximity to the representative ones. Considering k representative objects, the first genotype represents two clusters, the second genotype represents three clusters,..., and the last one represents k clusters. Thus, we have employed initial populations formed by $(k-1)$ genotypes, where each genotype represents a different clustering.

4. Simulations

This section describes the application of the Bayesian approach as a data preparation tool for the CGA. Basically, the simulations provide a way of evaluating the Bayesian substitution method in the context of the CGA. Thus, the efficacy of the proposed method is illustrated by means of the results obtained on three different datasets: *Ruspini*, 200 Randomly Generated and Wisconsin Breast Cancer.

The datasets *Ruspini* and *200 Randomly Generated* originally do not have missing values. Therefore, we randomly removed some values in order to simulate the missing ones. It was performed by removing, independently, 30% of the values from feature “x” and 30% of the values from feature “y”. Thus, we got four samples (two for *Ruspini* - substituting “x” and “y” respectively – as well as two for *200 Randomly Generated*). In addition, the continuous values were naive discretized. The random process used to remove 30% of the values is based on the “Minimal” random number generator [18].

Besides, we employed the Wisconsin Breast Cancer Database to show an application that *naturally* contains 16 objects with missing values. In this way, we got two datasets: one containing the 683 objects without missing values and other containing the 16 objects with substituted values.

In each simulation performed by the CGA, we employed populations formed by 20 genotypes that, in turn, implies in using 21 clusters at most. Besides, the maximum number of generations was set to 100. The Euclidean distance was used as the metric to calculate the dissimilarities between objects and we simulated 10 experiments on each dataset.

Our main goal in this work is to develop a method that combines the Bayesian network capability to substitute missing values with the CGA potential capability of producing classification rules. In this sense, in principle we are just interested in evaluating the substitution process in the context of the CGA. Thus, we do not intend to compare the classification results provided by Bayesian networks with those obtained by the CGA. However, it is useful to know the average classification rates obtained by the Bayesian networks used to substitute the missing values, because these results can show that Bayesian networks are suitable to substitute missing values in the datasets used in our simulations. To perform the classifications, the original datasets were divided into five equal size samples. In each experiment, 80% of the data was used as the training sample, and the remaining data formed the test sample. This process was performed five times, originating five experiments. The random number generator [18] was used to extract, without replacement, the objects from the original sample. Therefore, the experiments minimize the bias of classifying the same objects that were used in the training process.

4.1. Example1 – *Ruspini* Data

The first simulation deals with the well-known *Ruspini* data [17]. There are 75 objects described by means of two attributes $\{x, y\}$. Four groups form the right clustering, as it can be seen in Figure 2. These four groups are formed by 20, 23, 17 and 15 objects. Thus, the sample with artificially missing values is formed by 6, 7, 5 and 4 objects of these groups respectively, i.e. the sample is formed by 22 objects.

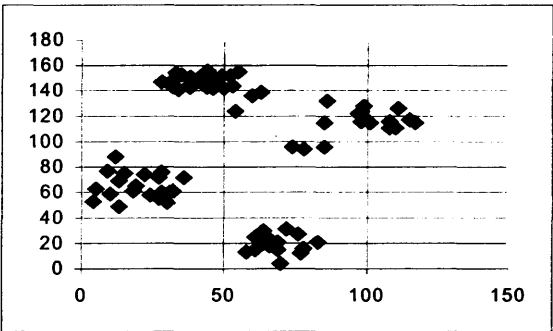


Figure 2. Ruspini Data.

Considering the complete dataset, the CGA always found the right clustering, on average after 11.9 generations (standard deviation=10.7), with an Objective Function Value (OFV) equals to 0.73. Considering the sample formed by the 22 objects whose “x” values were substituted, the CGA found 5 groups, i.e. it divided the group formed by 7 objects into two subsets and found the other ones correctly. This solution was found on average after 1.7 generations (standard deviation=1.3) and, although the CGA did not found the *right* clustering, the Average Classification Rate (ACR) was equal to 100%. When the substitution was performed to the y values, the CGA always found the *right* clustering in the initial population (OFV = 0.81).

To highlight the quality of the 22 missing values substitution process, we calculated the average distance from the discretized original values to the substituted ones. Considering the “x” feature, the average distance was equal to 1.05 in the range [0,12], and for the “y” feature this measure was equal to 0.6 in the range [0,16].

Considering the classification task using the whole dataset (without missing values), one observes that the Exhaustive Ordering Search (EOS) Bayesian classifier [19] found the right class in 86.93% of the cases (standard deviation = 3.49%). Working with the 22 completed objects whose “x” missing values were substituted, the EOS found the right class in all the objects and, working with objects whose “y” missing values were substituted the ACR was equal to 90.91%.

4.2. Example 2 – 200 Randomly Generated Objects

The second simulation deals with 200 objects described by two variables randomly generated [17]. Three groups of objects are constructed and in each group the points are generated according to a spherical bivariate normal distribution with given mean vector $\mu=(\mu_x,\mu_y)$ and standard deviation σ for both “x” and “y” according to:

Group1: 120 objects	$\mu_x = 0$	$\mu_y = 10$	$\sigma = 1.7$
Group2: 60 objects	$\mu_x = 20$	$\mu_y = 12$	$\sigma = 0.7$
Group3: 20 objects	$\mu_x = 10$	$\mu_y = 20$	$\sigma = 1.0$

The three groups depicted in Figure 3 form the right clustering. From these, we randomly selected 30% of each group (36, 18 and 6 respectively), obtaining a sample formed by 60 objects whose values of “x” and “y” were eliminated and consequently substituted.

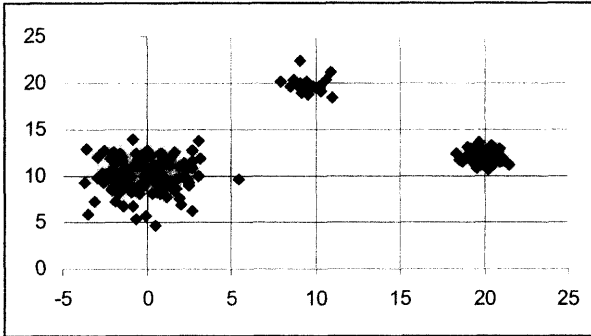


Figure 3. Randomly Generated Data.

Considering the complete dataset, the CGA found the right clustering always in the initial population ($OFV=0.82$). Considering the 60 objects whose x attributes were substituted, the CGA always found 9 groups in the initial population ($OFV=0.94$, $ACR=90\%$). Actually, the CGA did not correctly classify 5 objects of the subset formed by 36 objects as well as one object of the group formed by 18 objects. Thus, the ACRs for the groups formed by 36, 6 and 18 objects were equal to 86.11%, 100% and 94.44% respectively. When the substitution was based on the “ y ” values, the CGA always found 11 clusters in the initial population ($OFV=0.98$, $ACR=100\%$). In fact, it actually found 6 clusters in the group formed by 36 objects, 2 clusters in the group formed by 6 objects and 3 clusters in the group formed by 18 objects.

Again, we applied the same substitution process previously performed to the Ruspini dataset and the average distances from the discretized original values to the substituted ones were calculated. Considering the “ x ” feature, the average distance was equal to 3.08 in the range $[0,25]$, and for the “ y ” feature this measure was equal to 1.0 in the range $[0,17]$. Using the whole dataset (without missing values), the EOS found the right class in 99.5% of the objects (standard deviation = 1.11%). Besides, considering the 60 objects whose “ x ” missing values were substituted, the EOS found the right class in 78.33% of the cases, whereas in the objects whose “ y ” missing values were substituted, the EOS found the right class for all the objects.

4.3. Example 3 – Wisconsin Breast Cancer Data

This database is available at the UCI Machine Learning Repository [20]. Each object has 9 attributes and an associated class label (benign or malignant). The two classes are known to be linearly inseparable. The total number of objects is 699 (458 benign and 241 malignant), of which 16 have a single missing feature (14 of the benign class and 2 of the malignant class).

Considering the complete dataset, the CGA found two clusters in all simulations (in the average after 4.3 generations – standard deviation equals to 4.00), but the clusterings were not always the same ones. Thus, the ACR was not the same one in each experiment, i.e. it varied from 95.02% ($OFV=0.5970$) to 95.75% ($OFV=0.5969$), which is slightly better than the results described in [21]. Besides, one observes that the *right* clustering to this problem provides an OFV equals to 0.57, which is smaller than those obtained by the CGA. It happens because some objects have negative $s(i)$ values when these are calculated according to the groups formed by means of the class labels. In other words, some objects are more similar to the ones of the opposite class, what prevents the CGA from finding the *right* solution. Considering the benign examples, the ACR varied from

98.20% to 98.42%, whereas this rate varied from 88.70% to 91.21% when the malignant examples are considered. When the CGA was applied in the 16 objects, it always found the same solution (in the initial population), formed by two groups (OFV=0.64). The ACR was equal to 81.25% (85.71% in the benign class and 50% in the malignant class).

The missing values in this dataset were not artificially inserted. Therefore, it is not possible to measure the distance between the original values and the substituted ones. The Bayesian classification performed in the clean dataset (without missing values) found the right class in 95.12% of the objects and, using the sample formed by 16 objects the classification process found the right class for all the objects. It is worth to say that we have not used the EOS in this dataset yet, i.e. the original ordering of the attributes was employed.

4.4. Summary

This section describes the main results obtained in the performed simulations. Basically, it is possible to see (Table 1) that both the CGA and the Bayesian network provided good Average Classification Rates (ACRs) in the employed datasets. However, we are more interested in evaluating the efficacy of the Bayesian substitution method as a data preparation tool for the CGA. Thus, it is interesting to observe (Table 2) that the substitution method was consistent, i.e. the ACRs in the datasets formed by substituted objects were comparable to the ones obtained in the clean datasets. However, it is also possible to see that the ACR provided by the CGA in the substituted objects of the Wisconsin Breast Cancer dataset was smaller than the ACR found in the clean dataset. In this case, we believe that it happened because of the very low number of objects with missing values (only 2.29% of the dataset).

Table 1. Average Classification Rates (ACRs) - Complete datasets.

Complete Dataset	CGA Method	Bayesian Network Method
Ruspini	100.00%	86.93%
200 Randomly Generated	100.00%	99.50%
Wisconsin Breast Cancer	95.42%	95.12%

Table2. Average Classification Rates (ACRs) - Datasets with substituted missing values.

Dataset	CGA	Bayesian Network
Ruspini – 30% in “x”	100.00%	100.00%
Ruspini – 30% in “y”	100.00%	90.91%
200 Randomly Generated – 30% in “x”	90.00%	78.33%
200 Randomly Generated – 30% in “y”	100.00%	100.00%
Wisconsin Breast Cancer – 16 examples	81.25%	100.00%

5. Conclusions and Future Work

This work shows the application of a data mining method that combines a Bayesian data preparation approach with a Clustering Genetic Algorithm (CGA). In this sense, the K2 algorithm was used to substitute missing values in data preparation tasks. Our main goal was to evaluate the Bayesian substitution method in the context of some simulations performed by the CGA, which can provide useful rules for data mining applications. The obtained results in the three employed datasets show that the method is very promising. In this sense, the employed substitution process provided consistent results, in terms of the average classification rates, in the simulations performed by the CGA.

One important future work will be the application of the proposed method in other databases where the missing values problem is bigger. Besides, it is worth to investigate the application of the proposed method to extract prepositional classification rules. Another important future development consists in evaluating the performance of the employed Bayesian approach as a feature selection tool for the CGA. In this sense, we believe that it is possible to find a subset of the original features that provides simpler classification rules.

References

- [1] FAYYAD, U. M., SHAPIRO, G. P., SMYTH, P. "From Data Mining to Knowledge Discovery : An Overview". In: *Advances in Knowledge Discovery and Data Mining*, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., Editors, MIT Press, pp. 1-37, 1996.
- [2] BIGUS, J. P., *Data Mining with Neural Networks*, First edition, USA, McGraw-Hill, 1996.
- [3] HAN, J. & KAMBER, M. - *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.
- [4] HECKERMAN, D. - A tutorial on learning bayesian networks. *Technical Report MSR-TR-95-06*, Microsoft Research, Advanced Technology Division, Microsoft Corporation, 1995
- [5] HRUSCHKA Jr., E. R. & EBECKEN, N.F.F, Missing values prediction with K2. *Intelligent Data Analysis (IDA)*, Netherlands, v.6, n.6, to appear in 2002.
- [6] HECKERMAN, D. Bayesian Networks for Data Mining. *Data Mining and Knowledge Discovery Journal*, 1, 79-119, 1997.
- [7] David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rounthwaite, and Carl Kadie. *Dependency Networks for Inference, Collaborative Filtering, and Data Visualization*. Journal of Machine Learning Research Volume 1, pp. 49-75, 2000.
- [8] KEEDWELL, E., NARAYANAN, A., SAVIC, D. Creating Rules from Trained Neural Networks Using Genetic Algorithms, *International Journal of Computers, Systems and Signals (IJCSS)*, vol.1,n.1, pp. 30-42, December 2000.
- [9] HRUSCHKA, E. R., EBECKEN, N.F.F. A genetic algorithm for cluster analysis. *Intelligent Data Analysis (IDA)*, Netherlands, v.7, n.1, to appear in 2003.
- [10] DUCH, W., ADAMCZAK, R., GRABCZEWSKI, K. A New Methodology of Extraction, Optimization and Application of Crisp and Fuzzy Logical Rules, *IEEE Transactions on Neural Networks*, vol.11, n.2, pp.1-31, March 2000.
- [11] QUINLAN, J. R. Unknown Attribute Values in Induction. *Proceedings of 6th International Workshop on Machine Learning*, 164-168, Ithaca, NY, 1989.
- [12] COOPER G. & HERSKOVITZ, E. A Bayesian Method for the Induction of Probabilistic Networks from Data. *Machine Learning*, 9, 309-347, 1992..
- [13] FALKENAUER, E., *Genetic Algorithms and Grouping Problems*, John Wiley & Sons, 1998.
- [14] ARABIE, P., HUBERT, L. J., An Overview of Combinatorial Data Analysis (Chapter 1). *Clustering and Classification*, ed. P. Arabie, L.J. Hubert, G. DeSoete, World Scientific, 1999.
- [15] PARK, Y., SONG, M., "A Genetic Algorithm for Clustering Problems", *Proceedings of the Genetic Programming Conference*, University of Wisconsin, July, 1998.
- [16] COLE, R. M., *Clustering with Genetic Algorithms*, Master of Science Thesis, Department of Computer Science, University of Western Australia, 1998.
- [17] KAUFMAN, L., ROUSSEEUW, P. J., *Finding Groups in Data – An Introduction to Cluster Analysis*, Wiley Series in Probability and Mathematical Statistics, 1990.
- [18] PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T. & FLANNERY, B. P., *Numerical Recipes in C: The Art of Scientific Computing*. Second Edition, Cambridge University Press, 1992.
- [19] HRUSCHKA Jr., E. R. & EBECKEN, N.F.F, Ordering attributes for missing values prediction and data classification. *Proceedings of the Third International Conference on Data Mining Methods and Databases for Engineering, Finance and Other Fields*, to appear, 2002.
- [20] MERZ, C.J., MURPHY, P.M., UCI Repository of Machine Learning Databases, [http://www.ics.uci.edu]. Irvine, CA, University of California, Department of Information and Computer Science.
- [21] KOTHARI, R., PITTS, D., "On finding the number of clusters", *Pattern Recognition Letters*, n.20, pp. 405-416, Elsevier, 1999.

Reconstruction of conditional distribution field based on empirical data

Chervonenkis Alexey Jakovlevich
Royal Holloway University of London (London),
Institute of Control Science (Moscow)
Integra Ltd. (Moscow)

In the paper the problem of conditional distribution estimation at a current point on the basis of measured values of a random field at a set of sampling points is considered.

An estimator is searched in the form similar to Parsen estimation of distribution function, but with coefficients depending on the distance between the current point and the sampling points. Theoretical foundation is given for optimal choice of these coefficients. Applications of the theory are considered in the learning theory and practical problems.

1. Introduction

In many applications a problem arises of random field reconstruction using measurements made over some sampling net (random or regular). These applications include simulation of useful mineral grade field on the base of exploration sampling, environmental investigations, meteorology and many other tasks. The problem arises also in learning theory. Usually the problem is interpreted as evaluation of the field value at a current point using known values at the measurement points. But in many applications sufficiently precise field value estimation is impossible for the major part of its domain (it should be established after more detailed sampling). At the first stage of sampling it is enough to evaluated expected distribution of possible values at a current point. In more precise terms we have to evaluate conditional distribution of the field values at the current points given sampled values. There is known parametric technique to estimate conditional distribution. In particular for Gaussian field it is enough to find conditional mean and variance at the current point using Kriging technique [2,3]. Still in applications the field values distribution is often far from Gaussian. It was also proposed to use fixed data transform to reduce the distribution to the Gaussian case. But this approach also often fails, because uniform transform is used, while different transformation is necessary for different parts of the field domain. That is why it is natural to use non-parametric technique [1,4].

The most simple known way to estimate cumulative distribution function of a random value is construction of empirical cumulative histogram based on independent sampling sequence. Given the sequence of iid values c_1, c_2, \dots, c_n , the empirical cumulative histogram is estimated as:

$$(1) \quad F_c(t) = 1/n \sum I(t - c_i), \quad \text{where}$$

$$\begin{aligned} I(z) &= 0 & \text{for } z < 0, \\ I(z) &= 1 & \text{for } z \geq 0. \end{aligned}$$

Using more fine technique of Parsen method the distribution function is estimated similarly as:

$$(2) \quad F_c(t) = 1/n \sum S(t - c_i),$$

where $S(z)$ is some cumulative distribution function with zero mean and sufficiently small variance.

In our case it is natural to cumulate the histogram using the sampled values within some surrounding of a current point with weights depending on the distance between the current point and the points of sampling. Namely, if the values c_2, \dots, c_n are sampled at the points x_1, x_2, \dots, x_l then the conditional distribution function at the current point x is estimated as:

$$F_x(t) = \sum a(x, x_i) I(t - c_i), \quad \text{or}$$

$$F_x(t) = \sum a(x, x_i) S(t - c_i), \quad \text{where}$$

$\sum a(x, x_i) = 1$ and $S(z)$ is some cumulative distribution function with zero mean and sufficiently small variance..

The goal of the present work is to define in some sense optimal values of the weights $a(x, x_i)$.

2. Problem statement

For this purpose we introduce [1] a new notion of (generalized) random field defined over some metric space \mathbf{R} which values are (cumulative) distribution functions $F_x(c)$ ($x \in \mathbf{R}$) of real values c . The scheme of generation of a real (physical) field is as follows. There are two steps of generation. First step: at each point x of the space \mathbf{R} a distribution function $F_x(c)$ is generated according to our definition of generalized field. Second step: at each point a real field value is generated at random according to the distribution $F_x(c)$ independently from point to point. In particular this notion allows to describe common random fields, fields with uncorrelated noise and other. In general this approach is more convenient in the cases when the distance between sampling points is large in comparison with correlation radius of the random field (in common sense).

To be more precise let us denote \mathbf{F} the set of all possible one dimensional distributions $F(c)$. So each function $F(c)$ is a monotonous real function of a real variable, such that $0 \leq F(c) \leq 1$, $F(-\infty)=0$, $F(\infty)=1$, $F(c+)=F(c)$. Now let $(\Omega, \Sigma, \mathbf{P})$ be a probabilistic space, where Ω is the set of elementary events, Σ is the sigma-algebra of events and \mathbf{P} is probabilistic measure. \mathbf{R} is some metric space. We define a generalized random field as a function $F_x(c, \omega)$ ($x \in \mathbf{R}$, $\omega \in \Omega$) so that for each fixed x and ω the function $F_x(\cdot, \omega)$ belongs to the set \mathbf{F} , i.e. is a cumulative distribution function. Due to this definition $F_x(c, \cdot)$ is a random value (in common sense) for any $x \in \mathbf{R}$ and real value c . Further we shall denote expectations due to this definition as E_g .

To transfer from this definition to a random field in common sense we define a random scalar field $C(x)$ over $x \in \mathbf{R}$ by

$$(3) \quad P(C_{x1} \leq c_1, C_{x2} \leq c_2, \dots, C_{xn} \leq c_n) = E_g F_{x1}(c_1), F_{x2}(c_2) \dots, F_{xn}(c_n),$$

which corresponds to independent generation of values c for given realization of the generalized random field.

Our goal is given the field values $C_{x1}, C_{x2}, \dots, C_{xn}$ at the sampling points x_1, x_2, \dots, x_n to construct the optimal estimator for $F_y(c)$ (y is a current point in R) of the kind

$$(4) \quad F_{ey}(c) = \sum a(y, x_i) S(c - C_{xi}),$$

where

$$\sum_{i=1,n} a(y, x_i) = 1$$

and S is some fixed (cumulative) distribution function with zero mean (in the simplest case it is just $I(z)$ defined by (1)).

The criterion of the distance between the estimation and true distribution function (to be minimized) is defined as

$$(5) \quad D = E_g \int (F_{ey}(c) - F_y(c))^2 dc \rightarrow \min.$$

3. Problem solution

In geostatistics [2,5,6] the notion of variogram is widely used. For a random field $C(x)$ it is defined as a non-negative real function

$$V(x, y) = E (C(x) - C(y))^2.$$

The variogram is used as the basis for kriging equations. According to this definition $V(x, x) = 0$. But usually

$$\lim V(x, y) = V_0(x) > 0 \quad (x \rightarrow y).$$

The value $V_0(x)$ corresponds to the variance of uncorrelated component of the field at the point x . In geostatistics it is called nugget effect, reminding of gold nuggets as the uncorrelated component.

Also the so called **variogram of order one** is used defined as

$$(6) \quad V_1(x, y) = E |C(x) - C(y)|.$$

For generalized random field a generalized notion of variogram can be introduced defined as

$$(7) \quad V_g(x, y) = E_g \int (F_x(c) - F_y(c))^2 dc.$$

Usually $\lim V(x, y) = 0 \quad (x \rightarrow y)$, which means that distribution functions $F_x(c)$ and $F_y(c)$ are in average close for close points x and y , even if the real random field has uncorrelated component. It appears that the generalized "nugget effect" corresponds to the value

$$V_g^0(\mathbf{x}) = E_g \int F_x(c)(1 - F_x(c)) dc$$

which determines expected dispersion of the value c .

Further we consider generalized fields stationary in the following weak sense:

$$E_g F_x(c) \text{ and } E_g \int F_x(c)(1 - F_x(c)) dc = V_g^0$$

do not depend on \mathbf{x} .

It was shown in [1] that for stationary fields in the case $S(z) = I(z)$ coefficients $a(\mathbf{y}, \mathbf{x}_i)$ which deliver minimum to the criterion (5) can be found solving the system of linear equations

$$(8) \quad \mathbf{T}\mathbf{a} = \mathbf{r}$$

where the matrix \mathbf{T} ($(n+1) \times (n+1)$) has elements

$$\begin{aligned} t_{ii} &= 0, \\ t_{ij} &= V_g(\mathbf{x}_i, \mathbf{x}_j) + 2V_g^0 \quad \text{for } 1 \leq i, j \leq n, i \neq j, \\ t_{i, n+1} &= 1 \quad (1 \leq i \leq n), \\ t_{n+1, j} &= 1 \quad (1 \leq j \leq n), \end{aligned}$$

vector \mathbf{r} has components

$$\begin{aligned} r_i &= V_g(\mathbf{x}_i, \mathbf{y}) + 2V_g^0 \quad (1 \leq i \leq n), \\ r_{n+1} &= 1 \end{aligned}$$

and unknown vector \mathbf{a} is formed by the searched coefficients

$$a_i = a(\mathbf{y}, \mathbf{x}_i) \quad (1 \leq i \leq n),$$

and (unknown) Lagrangian multiplier

$$a_{n+1} = \lambda.$$

(The last row and column of the matrix \mathbf{T} form really Lagrangian function corresponding to the restriction $\sum_{i=1, n} a(\mathbf{y}, \mathbf{x}_i) = 1$).

In the general case when $S(z)$ is an arbitrary zero mean (cumulative) distribution the optimal coefficients $a(\mathbf{y}, \mathbf{x}_i)$ may be found solving equation system similar to (8). But the generalized variogram is calculated for transformed distribution functions. Each distribution function $F_x(c)$ of the generalized random field is transformed to

$$F_x^*(c) = \int S(c-t) dF_x(t).$$

It corresponds to “smoothing” initial distribution by Parzen window with density $p(z) = dS(z)/dz$, which in term is equivalent to adding independent zero mean noise distributed according to $S(z)$.

Now the generalized variogram is transformed to

$$V_g^*(x, y) = E_g \int (F_x^*(c) - F_y^*(c))^2 dc,$$

and the nugget effect is changed to

$$V_g^*(x) = E_g \int F_x^*(c)(1 - F_x^*(c)) dc - \int S(c)(1 - S(c)) dc.$$

The linear equation system (8) is changed to

$$(8^*) \quad T^* a = r^*,$$

with

$$t_{ij}^* = V_g^*(x_i, x_j) + 2V_g^* \quad \text{for } 1 \leq i, j \leq n, i \neq j,$$

and

$$\begin{aligned} r_i^* &= V_g^*(x_i, y) + 2V_g^* \quad (1 \leq i \leq n), \\ r_{n+1}^* &= 1 \end{aligned}$$

The diagonal of the matrix T and boundary row and column remain the same as in (8).

4. Reduction to the variogram of order 1

It can be shown that the generalized variogram $V_g(x, y)$ defined by (7) can be expressed in terms of variogram of order 1 for the random field in common sense. It is due to the following trivial (but not evident) equality.

Let u and t be independent random variables distributed according to distribution functions $F_1(u)$ and $F_2(t)$. Then

$$(9) \quad E|u - t| = \int (F_1(c) - F_2(c))^2 dc + \int F_1(c)(1 - F_1(c)) dc + \int F_2(c)(1 - F_2(c)) dc.$$

Really

$$E|u - t| = E \int [I(c - u) - I(c - t)]^2 dc,$$

and as far as $I^2(z) = I(z)$

$$\begin{aligned} E|u - t| &= E \int [I(c - u) + I(c - t) - 2I(c - u)I(c - t)] dc = \\ &= \int [E I(c - u) + E I(c - t) - 2E I(c - u)E I(c - t)] dc. \end{aligned}$$

Now according to definition

$$E I(c - u) = F_1(c) \quad \text{and} \quad E I(c - t) = F_2(c).$$

Then

$$\begin{aligned} E|u - t| &= \int (F_1(c) + F_1(c) + 2F_1(c)F_2(c)) dc = \\ &= \int (F_1(c) - F_2(c))^2 dc + \int F_1(c)(1 - F_1(c)) dc + \int F_2(c)(1 - F_2(c)) dc, \end{aligned}$$

and we have the result (9). All the integrals here are finite if and only if the random values u and t have finite expectations.

The first item of the sum reflects the distance between distributions, the next items are equal to the so called dispersion of each distribution. The result seems rather strange, because if we calculate $E(u-t)^2$ instead of $E|u-t|$ the result reflects only the distance between average values of the two distributions, but not the distance between distributions themselves.

Now we can express the variogram of order 1 in terms of generalized random field. According to definition (6)

$$V_1(\mathbf{x}, \mathbf{x}) = 0,$$

$$V_1(\mathbf{x}, \mathbf{y}) = E |C(\mathbf{x}) - C(\mathbf{y})| = E_g E_c |C(\mathbf{x}) - C(\mathbf{y})|, \text{ for } \mathbf{x} \neq \mathbf{y},$$

where the symbol E_c means expectation for two independent values $C(\mathbf{x})$ and $C(\mathbf{y})$ distributed according to some fixed distributions $F_x(c)$ and $F_y(c)$. Now using the result (9) we get for $\mathbf{x} \neq \mathbf{y}$

$$V_1(\mathbf{x}, \mathbf{y}) = E_g \int (F_x(c) - F_y(c))^2 dc + E_g \int (F_x(c)(1 - F_x(c)) + F_y(c)(1 - F_y(c))) dc.$$

Or in our terms

$$V_1(\mathbf{x}, \mathbf{x}) = 0,$$

$$V_1(\mathbf{x}, \mathbf{y}) = V_g(\mathbf{x}, \mathbf{y}) + V_g^0(\mathbf{x}) + V_g^0(\mathbf{y}), \text{ for } \mathbf{x} \neq \mathbf{y},$$

But these are just the values used to construct the matrix \mathbf{T} and the vector \mathbf{r} for equation system (8). So we can write for $V_g^0(\mathbf{x}) = \text{const}$

$$t_{ij} = V_1(\mathbf{x}_i, \mathbf{x}_j) \text{ for } 1 \leq i, j \leq n,$$

$$r_i = V_1(\mathbf{x}_i, \mathbf{y}).$$

Thus the problem can be solved using only first order variogram of a random field in common sense. So in this most simple case the result is close to that proposed in 1983 by A. Journel and called indicator kriging.

In the case when $S(z)$ is an arbitrary zero mean (cumulative) distribution the variogram $V_g(\mathbf{x}, \mathbf{y})$ is to be changed to

$$V_g^*(\mathbf{x}, \mathbf{y}) = E_g \int (F_x^*(c) - F_y^*(c))^2 dc.$$

As it was mentioned above the transform of the generalized field $F_x(c)$ is changed to

$$F_x^*(c) = \int S(c-t) dF_x(t),$$

which is equivalent to adding independent noise distributed according to $S(z)$. So according to the result (9)

$$V_g^*(\mathbf{x}, \mathbf{y}) + 2V_g^* = E |C^*(\mathbf{x}) - C^*(\mathbf{y})| - V^{**},$$

where $C^*(x) = C(x) + \xi$ (ξ is independent noise distributed according to $S(z)$) and $V^{**} = \int S(c)(1 - S(c)) dc$.

By substitution of these values to the matrix T^* and the right side r^* we get the equation system to find optimal coefficients $a(y, x_i)$ in the general case. This result seems to be new.

5. Applications

A software complex for computer simulation of large ore deposits based on exploration data was developed by **Integra** company (Moscow) using these ideas.

The whole volume of the bowels of the earth containing a deposit is splinted to a set of rectangular cells. Then for each cell the conditional distribution of useful mineral grades (within the cell) is estimated based on exploration sampling data. The 3D system of the cells with assigned conditional distribution forms the deposit model. Using this model it is possible not only to estimate the resources of the deposit as the whole or any part of it but also to carry out its development scheduling. Geological boundaries are estimated using different kinds of exploration data. This boundaries serve to restrict interpolation within geologically homogenios zones of a deposit. Logical inference, regression technique and pattern recognition are used to estimate the position of the boundaries and spatial variogram parameters. So the complex can be considered as a hybrid system.

The software complex was applied for modeling of a set of large gold, copper and nickel deposits in Russia and other countries of CIS. In particular it was used for computer simulation and long term scheduling of the world largest gold open pit Muruntau (Uzbekistan). Using blast hole sampling (much more dense than that used for the deposit modeling) it was possible to compare block resource estimation made in traditional way with that based on our model. The values calculated on blast hole sampling were accepted as the true ones. The table below shows average relative deviation of estimates from the true values for different cut off grades.

Cut off grade, g/t	Estimation based on the model			Estimation based on traditional (hand made) calculation		
	Ore	Grade	Metal	Ore	Grade	Metal
1,0%	9,5%	9,4%	14,8%	20,7%	12,7%	23,2%
1,5%	11,9%	8,8%	17,8%	22,4%	13,8%	26,3%
2,0%	17,5%	10,1%	22,7%	28,0%	14,3%	29,8%

This approach can be applied also to different machine learning problems, when it is necessary to estimate not the value of some item of a new object, but the expected distribution of the value.

References.

[1] Ju. Ju. Bakhtin, A.V. Danilov, A.V. Kantsel. *Method for reconstruction of conditional distributions based on empirical data.* (Rus.) Avtomatika i telemekhanika. V. 12, 2000, Moscow. English translation: Automation and remote control.

- [2] Jean-Paul Chiles, Pierre Delfiner. **Geostatistics: Modeling Spatial Uncertainty**. A Wiley-Interscience Publication, 1999.
- [3] Journel A.G., Huijbregts C.G. *Mining Geostatistics*, Academic Press, London, 1978.
- [4] Journel A.G., Johnson T.B., Barnes R.J. The indicator approach to estimation of spatial distributions. In *Proceedings of the 17th APCOM International Symposium*, New York, 1983.
- [5] Journel A.G. Nonparametric estimation of spatial distributions. *Journal of the International Association for Mathematical Geology*, N 15 (3), 1983.
- [6] Matheron G., Verly G., David M., Journel A.G., Marechal A. The selectivity of the distributions and the "second principle of geostatistics". In *Geostatistics for Natural Resources Characterization, Part 1*, 1984.

Bi-Directional Flow of Information in the Softboard Architecture

Silvio MACEDO and Ebrahim MAMDANI

*Intelligent & Interactive Systems, Department of Electrical and Electronic Engineering,
Imperial College London,
Exhibition Road, London SW7 2BT, England.*

Abstract. This paper addresses the issue of integrating top-down and bottom-up processing of information in modular hybrid systems. Backed by computational and neurological evidence, we show that there are important advantages to be gained from supporting bi-directional flow of information, especially in the context of intelligent information processing in real-world applications. We consider modular hybrid systems from a granular computing perspective and propose an approach based on evidential reasoning to integrate both top-down and bottom-up processes. The implementation of the model in the Softboard framework, an experimental distributed hybrid architecture, is presented and its application to intelligent filtering and retrieval of multimedia is illustrated.

1. Introduction

Information processing is at the core of most complex real-world applications. As articulated by Sun in [1], no single problem-solving methodology is able to cope with all aspects of such applications. Thus, hybrid systems, which aim to combine the strengths of individual techniques, have been receiving increased attention.

In the majority of the cases, computers are programmed to process information in either a dominant bottom-up, or a dominant top-down fashion. Despite their power, top-down approaches have pitfalls when applied to real-world scenarios, especially in terms of their sensitivity to noise, limited learning capability [1] and symbol-grounding [2]. On the other hand, bottom-up information processing tends to be limited in terms of the complexity of operations it can perform.

To overcome these problems, the hybridisation or fusion of symbolic and connectionist techniques, in a predominantly bottom-up arrangement, is tried in hybrid systems. This integration may take different forms, and can be classified in different ways, as in [3-6]. However, if information flows predominantly in a bottom-up direction, much of the reasoning power of the symbolic components will be left untapped.

In the current work, we argue that to take further advantage of the synergies of combining the various processing techniques, attention must be given to the integration of both bottom-up and top-down reasoning and information processing. Information must flow bi-directionally, in the form of a continuous loop between existing knowledge and incoming information, at each pair of interacting modules, and between abstraction levels.

The structure of this paper is as follows. In Section 2 we refer to results from a range of research fields on the importance of bi-directionality for intelligent information processing, and discuss its potential advantages in the context of hybrid systems. In Section 3 we begin

by considering modular hybrid systems from a granular computing perspective, and based on this view, we offer a definition of bi-directional flow of information. Then, a model for top-down/bottom-up integration based on evidential reasoning is put forward. In Section 4, we present an implementation of this model in the Softboard framework and illustrate its application to filtering and retrieval of multimedia content. Section 5 concludes underlining the essence of our contribution and presents plans for future developments.

2. Bi-Directionality in Information Processing

In this section we study the potential advantages of bi-directional interaction between heterogeneous modules within a hybrid system and we hypothesize as to the qualitative improvements in reasoning power that may emerge from such interaction. Support is adduced from a broad base of sources, including neurological, cognitive and computer sciences.

2.1 *Bi-Directionality in Cognitive and Computational Models*

Corroborated by ample psychophysical and neuroanatomical evidence, bi-directionality is an essential part of several models of cognition. In [7], Mumford proposes a computational model of the cortex based on Pattern Theory, where bi-directionality is the essence of the proposed analysis-synthesis loop. In a related work, Rao and Ballard [8] achieve similar functionality, using a form of extended hierarchical Kalman filter. Ullman's sequence-seeking and counter-streams [9] offer yet another powerful alternative, in which bi-directionality is achieved by a process of priming among top-down and bottom-up pathways. In [10], evidence resulting from the analysis of somato-dendritic interactions in the brain, inspired Siegel et al. to model the integration of bottom-up and top-down information processing down to a cellular level.

In computer information processing systems, integration of top-down and bottom-up processes has been proposed in almost all kinds of computational paradigms. Carpenter and Grossberg's Adaptive Resonance Theory for self organizing neural networks [11] and its many derivatives are one of the best examples. But one also finds bi-directionality implemented in coupled hidden Markov models [12], in Helmholtz machines [13], in planning [14], in theorem proving [15], in abduction [16, 17], among many others.

The cognitive models referenced previously try to be biologically plausible and therefore some similarities between approaches can be drawn. In contrast, in the other examples given, the specific techniques vary greatly in each case. In result, information on the resulting improvements is quite scattered. In the next subsection we combine these results and elaborate on some of the potential benefits that bi-directionality may offer.

2.2 *Advantages of Bi-directionality*

From all the architectures and results discussed in the literature, it is undisputed that bi-directionality is generally considered a good, and in many cases, essential property in information processing systems. In this subsection, we explore some of the advantages that may emerge from bi-directionality.

Increased *performance*. As pointed out by Josephson [16], Ullman [9] and Fuchs [15], top-down processes may significantly cut-down the size of the search space of lower abstraction processes. In fact, top-down knowledge may effectively render some expensive lower-level computations superfluous altogether, and selectively activate processing modules and probe information sources [18]. Also, in a process similar to what is known in

computer processor design as look-ahead speculative processing, especially if processing in the modular system is distributed across machines, knowledge at different levels of abstraction may be processed in parallel, and in advance. After a new result emerges from one module, entire reasoning paths may be ready to immediately offer further results. Finally, if information can bi-directionally traverse layers of abstraction, performance gains may also be expected from dealing with information at the most appropriate level of abstraction. For example, if a speech recognition system is trying to ascertain the familiarity of a certain sound and if an almost perfect mapping between the audio input and some phonetic description was found, the system would execute faster if that representation was used instead of the raw audio.

Achieving these improvements in performance is not without a price. In this case, the proposed mechanisms are mostly based on the idea that various processing tasks can be processed in a specific useful order, or simultaneously. In turn, this requires some form of coordination mechanisms or some constraints on the validity of information at any given time, which raises the complexity of the system. Furthermore, it is worth noting that a system adopting these techniques will potentially take less time to get to a solution. Yet, this does not necessarily mean it will use less overall processor time. A lot of processing power may be spent in activities that lead to irrelevant results. Therefore, to fully exploit the proposed advantages, distribution of the hybrid processing modules across machines is advisable, as for example is proposed in Section 4.

Increased robustness. The bottom-up flow of information allows the creation of context at higher levels. This context may in turn be used to originate an inverse flow of information, in the form of top-down expectations. These expectations increase robustness by filling up some of the voids left empty by the lack of bottom-up information and corroborating uncertain information therefore potentially reinforcing the signal and filtering out noise. In a similar fashion, top-down information will also have a role in driving attention in input stages to seek evidence that will solve a conflict or disambiguate higher level competing hypotheses, as explained in [16]. Nevertheless this process may raise a stability concern. The loop of data→context→expectations→data may be unstable and generate an unbounded number of paths to be explored by the system. Thus, a way of controlling the relevance and meaningfulness of the information circulating in the system is required. A possible solution is the use of evidential theory to attribute degrees of belief and uncertainty to information. In this way, unless corroborating data is encountered, uncertainty increases at each iteration. As uncertainty increases, information value decreases, until the loop “fades away”. On the other hand, if evidence and other hypotheses are found to be coherent, the consensus operator of evidential reasoning will result in a decrease of uncertainty and the system will converge.

Improved learning. In most hybrid systems, learning is done at the module level. Yet, enforcing bi-directionality in the system opens the opportunity for a new dynamic of learning. Namely, system-wide loops of information, as opposed to output to input feedback connections in a single module, may provide the required labelling, error feedback or reward, for learning algorithms to learn in an unsupervised context. In the same way, learning techniques that are not able to learn on-line, could be occasionally switched off-line for improving their fitness to new data, replaying previously encountered situations, for which an adequate solution was found via another path. For example, consider Figure 1. Assume that module D is a neural network, and that the path $(F,G,H) \leftrightarrow D \leftrightarrow B$ may generally give faster results than an alternative path $(E,F) \leftrightarrow C \leftrightarrow B$, where module C is for example a rule based module. Module D may execute faster, but it requires off-line learning. Under the current proposal, the flows $(E,F) \leftrightarrow C \leftrightarrow B \leftrightarrow D$ and $(F,G,H) \leftrightarrow D$, would then provide the required training sets, virtually on-line, to train D off-line.

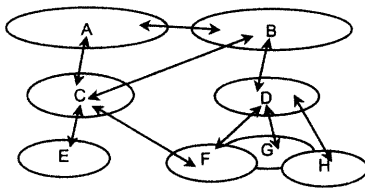


Figure 1. Template of the kind of information flow we propose in hybrid systems: a hierarchical composition, with multiple parallel, bi-directional flows of information.

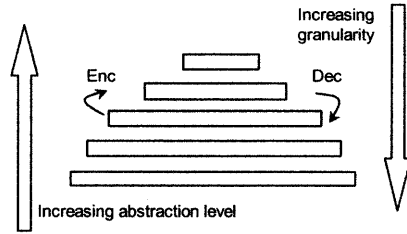


Figure 2. Information processing pyramid, suggesting that granularity decreases as abstraction level increases.

Enhanced *reasoning* skills. The processes described above may also result in an improvement of the reasoning capabilities of the system. For example, in case a module to solve a certain specific problem is missing, alternative paths leading to the same results may exist. In fact, the complex dynamics of interaction may cause the emergence of certain skills, which were not originally part of the system, and would not be possible without this sort of interaction. This is argued by Wegner [19], who follows a line of research that may in time become more relevant to the field of hybrid systems, and has at its core the interesting claim that “interaction is more powerful than algorithms”.

3. An Evidential Reasoning Model for Bi-directional Information Processing

In this section, we propose a form of bi-directional interaction based on Jøsang’s theory of Subjective Evidential Reasoning [20] and inspired on Josephson’s work on layered abduction [16]. Motivated by the need to generically address issues that are common to a certain class of actively coupled hybrid systems, as defined in [6], we begin by proposing that this class of hybrid systems be considered from a Granular Computing perspective. We then use this formalism to present a definition of bi-directionality in that context.

3.1 A Granular Computing Perspective of Multi-Module Hybrid Systems

The concept of granular computation was first introduced by Zadeh in [21]. Zadeh [22], Yao [23] and Pedrycz [24] give interesting introductory perspectives on the topic.

Granular Computing (GrC) deals with the representation and processing of chunks of information. Encoding and decoding operations are performed on these chunks, called information granules, granulating or generalizing them into other granules of different levels of abstraction and granularity, as generically represented in Figure 2.

It is interesting to find levels of abstraction as one of the main components of a purely computational model about which Zadeh underlines “... *its centrality to human reasoning*...” [22]. Siegel et al. [10], while working with basis on physiological evidence of cortical activity, have also pointed out that the presence of levels of abstraction are a common feature in the cognitive models of the cortex in the works of Mumford, Ullman, and Rao and Balard, among others. In our opinion, this should be seen as a strong indication that modular hybrid systems will also benefit from adopting an abstraction hierarchy. In what follows, we assume that is the case.

In the following, we use GrC concepts to describe a generic architecture for hierarchical multi-modular hybrid systems, as represented in Figure 1. These are the fundamental elements in such architecture:

- Information granules – information in the system, data or knowledge, exists in the form of granules (not shown in Figure 1). The specific definition of information granules in the system will depend on the application. Moreover, in opposition with most examples of GrC systems, the GrC relations of indistinguishability defining the partitions of the universe will depend on the various modules present in the system.
- Levels of abstraction – information granules are associated with a level of abstraction, based on their content and on the operations that can be performed on them.
- Processing modules – the encoding and decoding operations in granular computing, correspond in this architecture to the granulated “input” \leftrightarrow “output” functions executed by modules, on a single, or between any two abstraction levels. In the scenario of a hybrid system, there will generally be several processing modules, each module generating a coexistent valid partition of the universe.

3.2 A GrC-Based Definition of Bi-Directionality

Based on this analysis, and on Pedrycz formulation of GrC generic properties in [24], we now present a simple definition of bi-directional processing of information based on the generic principles of granular computing. We first introduce the notion of bi-directionality in relation to a single module.

Assume X and Y to be information granules. Take $ENC_M(X)$ as the encoding operation of X by module M , and $DEC_N(Y)$ as the decoding of Y by module N .

Definition 1. A module M is *strictly bi-directional* iff:

$ENC_M(X)$ and $DEC_M(X')$ are both defined, and
 $DEC_M(ENC_M(X)) = X$

That is, a module M is strictly bi-directional if, in addition to the normal input \rightarrow output operation, it is also able of processing its outputs and re-generating the original inputs. This exact bijective relation will frequently not hold or be at all possible. Yet, non strict bi-directionality is still worth pursuing and a non strict form of the definition can be relaxed to $\|DEC_M(ENC_M(X)) - X\| < \delta$, and δ is small, and $\| \cdot \|$ is an appropriate measure of distance.

Even if the above is not true about a module, if the module is able to use information in the output to change its behaviour, it may still have an important role in the bi-directionality of the system. This suggests the notion of a functionally bi-directional module. If we make $ENC_M(X|Y)$ to be the encoding of information granule X by module M , given that granule Y is present in the output space of M , functional bi-directionality can be defined as:

Definition 2. A module M is *functionally bi-directional* iff:

For Y, Z , granules on the output space of module M ,
 $Y \neq Z \Rightarrow ENC_M(X|Y) \neq ENC_M(X|Z)$

The value of functional bi-directionality is that it attributes to a module the property of changing its processing, in result of data at its output, or previous operations. This may happen due to a variety of reasons, the most important of which being that the module has some sort of learning, memory, or context awareness.

We now focus on the idea of bi-directionality at system level. Let Φ be a set of modules and $PATH_\Phi(X)$ be any nested sequence of operations performed on X , by modules in Φ . Finally, let L_i, L_j represent any two abstractions levels.

Definition 3. A hybrid system S is strictly bi-directional iff:
 for any pair of granules $X \in L_i$ and $Y \in L_j$,
 there is a set of modules $\Phi = \{M_1, M_2, \dots\}$ such that $Y = \text{PATH}_\Phi(X)$

Figure 1 is an example of such system, as it is possible to find a bi-directional path between any two modules, and therefore, any two granules at their inputs or outputs. In practice, making a system strictly bi-directional is difficult, and in many cases, the benefits would not compensate for the increased complexity. Hence, in an equally interesting sense, a system may be considered bi-directional, although not strictly so, if the system is bi-directional according to Definition 3 for a *significant* sub-set of abstraction levels, and all its modules are either bi-directional or functionally bi-directional. That is, there is always a mechanism by which the results of top-down processes influence the processing of bottom-up processes, and vice-versa.

3.3 Evidential Reasoning Based Bi-directionality

The previous two subsections created the setting for analysing hybrid systems using GrC concepts. Now, we make use of the established relation to put forward evidential reasoning, more specifically Jøsang's Subjective Logic [20], as an approach to implement bi-directionality.

Audun Jøsang's theory of subjective evidential reasoning [20, 25] defines a framework that allows to combine and assess subjective evidence from different sources. The operators defined in the corresponding Subjective Logic offer well-founded, powerful and intuitive methods to handle uncertainty, especially in chains of processing (i.e. a sequence of operations applied to one or more granules), as is prone to happen in our system. Although it builds on Dempster-Shafer belief theory, Jøsang's consensus operator does not suffer from the problems that affect both normalized and non-normalized versions of the Dempster consensus rule. For details and mathematical foundations we refer the reader to [25]. In this section we limit ourselves to describe how it is used in Softboard.

Using the Subjective Logic's concept of opinion and its discount and consensus operators, and given the GrC context presented, implementing bi-directionality is straightforward, just by defining a new type of information granule:

Definition 4. An *evidential-granule* is a n-tuple of the form $E = (I, O, M, L, T)$, where I is the specific information (data or knowledge) being communicated, O is the opinion, as defined in Subjective Logic, M is the module responsible for releasing this information, L is a list of links to other information granules, from which this granule originated, and T is the time at which the granule was produced.

In this fashion, we convert the problem of integration of top-down and bottom-up processes into one of combination of evidence, using the consensus operator. Each module will now operate on granules as evidence of a certain fact. Different degrees of belief, disbelief and uncertainty in the opinion will make the granule act as a hypothesis or as an item of evidence. By maintaining the links L , the dynamics of hypothesis testing, expectation and integration of bottom-up and top-down processes, as proposed in Josephson's Multi-Layered Abduction [16], become easily implementable. In addition, due to the use of a theory of evidential reasoning in replacement for an ad-hoc approach to evaluate hypothesis, those dynamics are also better justified.

4. Implementation of Bi-directionality in the Softboard Architecture and Its Application to Intelligent Access to Multimedia

Our proposal is implemented in a framework under development at our group, called Softboard. Softboard is a distributed platform to build multi-modular hybrid information processing systems. This work is part of ongoing research to explore issues in the area of intelligent information filtering and retrieval. In particular, we are interested in increasing the intelligence of systems that already provide an integrated and user-friendly interaction, as in Stathis et al. [26].

Our current application domain is the filtering and retrieval of MPEG-7 [27] indexed content. The MPEG-7 standard, formally named “Multimedia Content Description Interface”, provides a rich set of standardised tools to describe multimedia content, and has also some support for representation of user, and device profiles. MPEG-7 standardises both low-level, feature oriented, and high-level, semantic oriented descriptors. An overview on MPEG-7 is given by Pereira in [28]. In the context of this paper it suffices to say that MPEG-7 does not specify how this metadata is created or how it is to be used. We are primarily concerned with the later.

In the following subsection, we give some details on the implementation of Softboard and make brief references on how we are applying it to the analysis of MPEG-7 metadata.

4.1 Softboard Architecture and Its Application to MPEG-7

The architecture of Softboard, illustrated in Figure 3, follows the generic structure proposed in Section 3.1. Softboard is implemented in Java, as a fine-grained multi-component distributed system that resembles, in some aspects, blackboard architectures [29]. Softboard adopts an abstraction hierarchy similar to that of blackboard architectures, by supporting a hierarchical composition of the contributing modules, with several *shared mediums* for communication. The abstraction hierarchy is implemented through a series of alternating structures: Information Spaces (IS) and Execution Spaces (ES). Each individual IS or ES is a blackboard. Each blackboard is a Linda tuple-space [30] running on an IBM TupleSpace server [31]. IBM TupleSpaces is a framework which implements a transactional, fully distributed (Java RMI-based) Linda tuple space, together with a few extensions to the normal Linda operators. One of such extensions is the low-level support for XML Tuples, with which we implement our evidential information granules. This is particularly convenient, as MPEG-7 also uses XML, thus simplifying the implementation.

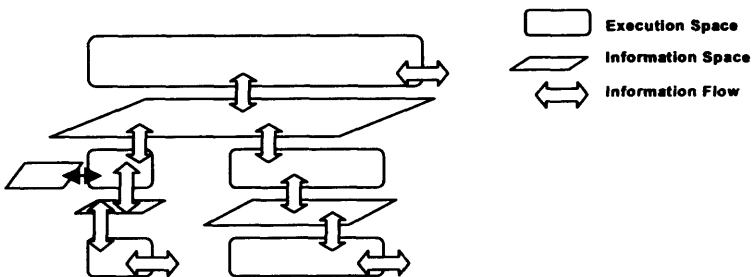


Figure 3. An example of a Softboard architecture.

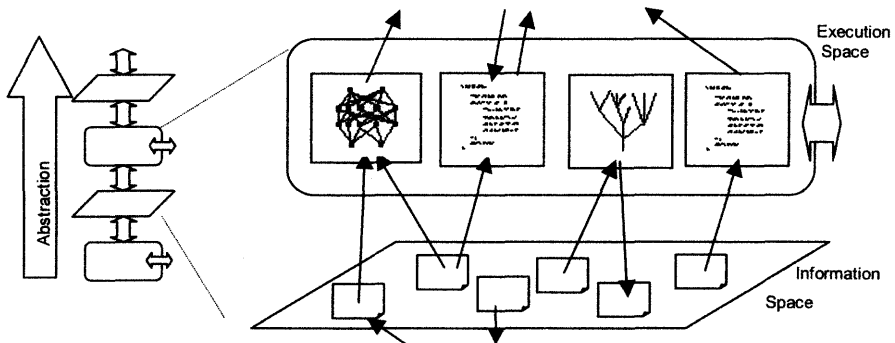


Figure 4. Diagram of a typical layer in the proposed system. The hybrid modules execute in parallel, reading and writing information from and to other layers of abstraction.

As exemplified in Figure 4, an IS is populated by evidential information granules. Depending on the specific abstraction level, these may be low-level features like colour histograms, or high level data, like target audience. ESs are populated by processing modules which implement the different algorithms, e.g. a scene-cut counter, or a text search in specific types of annotations. Each of these modules can be a thread or a completely separate process, running locally or remotely, and can read and write from any number of ISs, as well as to input/output information from/to the outside world. The modules are distributed by the various ESs according to the abstraction level of the task they execute in the context of the overall application.

Each Softboard application has an XML representation of its structure and components. A W3C XML Schema was designed to enforce validity of the structure and simplify the assembly of new Softboard applications, requiring nothing more than a simple, schema-aware visual XML editor. The defined XML schema was also used to automatically generate some of the Java classes, thus allowing human-readable and editable serialization of the state of the system in XML; something we find very convenient for debugging and analysis.

Support for evidential based bi-directionality is given in the platform in the form of Java classes that implement, and hide the details of extracting opinions from tuples and applying Subjective Logic calculus. In the construction of the modules, one has only to specify which other modules the algorithm depends on, and what type of tuples it requires access.

As an example, a neural network module which get its inputs from a “dark colour histogram” detector (directly from the MPEG-7 data), a scene-cut counter (through analysis of the content structure, in the MPEG-7 metadata) and movement-vectors information (from the actual MPEG-2 video stream) can be trained, using high level content descriptors, to identify certain type of movies, and then used when these are not available.

5. Conclusion

Based on other works in cognitive and computer science, we argued that in addition to combining different information processing paradigms, there are important advantages to be gained from also integrating top-down and bottom-up information processing. These potential advantages include increases in performance and robustness, and improved learning and reasoning capabilities.

Granular Computing was suggested as a tool to formalise and analyse hybrid systems.

Using this approach, we offered three definitions of bi-directionality, at module and system levels. More importantly, this bridge between GrC and hybrid systems allowed us to propose a methodology based on evidential reasoning and layered abduction, which transforms the problem of integration of bottom-up/top-down processes in one of combination of evidence.

The Softboard framework was briefly introduced and its application to information retrieval illustrated. Softboard was successful in allowing us to implement the proposed methodology for bi-directionality and the use of diverse information processing approaches. Yet, modular hybrid systems raise issues of knowledge representation and transformation between different hybrid components and levels for which Softboard does not yet offer a solution, and which are obviously non trivial. Future research will address this problem and consider solutions in the field of soft computing.

Acknowledgements

This work has been funded by the PRAXIS grants program of the Portuguese Science Foundation (FCT), which is gratefully acknowledged. We would also like to thank our colleagues Patrick Purcell, Yiannis Demiris and Oscar de Bruijn for very interesting and enlightening discussions, as well as some detailed comments on the manuscript, some on earlier versions.

References

- [1] R. Sun, "Artificial intelligence: connectionist and symbolic approaches," in *International Encyclopedia of Social and Behavioral Sciences*, vol. 2: Elsevier Science Ltd., 2001, pp. 783-789.
- [2] R. Sun, "Symbol grounding: a new look at an old idea," *Philosophical Psychology*, vol. 13, pp. 149-172, 2000.
- [3] M. Hilario, "Architectures and techniques for knowledge-based neuro-computing," in *Knowledge-Based Neurocomputing*, I. Cloete and J. M. Zurada, Eds. Cambridge, UK: MIT Press, 2000, pp. 27-62.
- [4] A. Browne and R. Sun, "Connectionist inference models," *Neural Networks*, vol. 14, pp. 1331-1355, 2001.
- [5] S. Wermter and R. Sun, *Hybrid neural systems*, vol. 1778. New York, NY, USA: Springer-Verlag Inc., 2000.
- [6] K. McGarry, S. Wermter, and J. MacIntyre, "Hybrid neural systems: From simple coupling to fully integrated neural networks," *Neural Computing Surveys*, vol. 2, pp. 62-93, 1999.
- [7] D. Mumford, "Neuronal Architectures for Pattern-theoretic Problems," in *Large-Scale Neuronal Theories of the Brain*, C. Koch and J. L. Davis, Eds. Cambridge MA: MIT Press, 1994, pp. 125-152.
- [8] R. P. N. Rao and D. H. Ballard, "Dynamic Model of Visual Recognition Predicts Neural Response Properties in the Visual Cortex," *Neural Computation*, vol. 9, pp. 721-763, 1997.
- [9] S. Ullman, *High-Level Vision : Object Recognition and Visual Cognition*. Cambridge, Mass.: MIT Press, 1996.
- [10] M. Siegel, K. P. Kording, and P. Konig, "Integrating Top-Down and Bottom-Up Sensory Processing by Somato-Dendritic Interactions," *Journal of Computational Neuroscience*, vol. 8, pp. 161-173, 2000.
- [11] G. A. Carpenter and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern-Recognition Machine," *Computer Vision Graphics and Image Processing*, vol. 37, pp. 54-115, 1987.
- [12] N. M. Oliver, B. Rosario, and A. P. Pentland, "A Bayesian Computer Vision System for Modeling Human Interactions," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 22, pp. 831-843, 2000.
- [13] P. Dayan and G. E. Hinton, "Varieties of Helmholtz Machine," in *Four Major Hypotheses In Neuroscience*, vol. 9; Number 8, *Neural Networks -Oxford-*, K. Toyama, N. Sugie, and M. Kawato, Eds.: Pergamon Press, 1996, pp. 1385-1404.
- [14] M. Veloso, J. Carbonell, A. Perez, and D. Borrajo, "Integrating planning and learning: the Prodigy

- architecture," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 7, pp. 81, 1995.
- [15] D. Fuchs and M. Fuchs, "Cooperation between top-down and bottom-up theorem provers," *Journal of Artificial Intelligence Research*, vol. 10, pp. 169-198, 1999.
 - [16] J. R. Josephson and S. G. Josephson, *Abductive Inference: Computation, Philosophy, Technology*. Cambridge, UK: Cambridge University Press, 1994.
 - [17] M. P. Shanahan, "A Logical Account of Perception Incorporating Feedback and Expectation," presented at Knowledge Representation, 2002.
 - [18] A. A. Salah, E. Alpaydin, and L. Akarun, "A Selective Attention-Based Method for Visual Pattern Recognition with Application to Handwritten Digit Recognition and Face Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence Pami*, vol. 24, pp. 420-424, 2002.
 - [19] P. Wegner, "Interactive Foundations of Computing," *Theoretical Computer Science*, vol. 192, pp. 315-351, 1998.
 - [20] A. Josang, "Subjective Evidential Reasoning," presented at 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2002), Annecy, France, 2002.
 - [21] L. Zadeh, "Fuzzy Sets and Information Granularity," in *Advances in Fuzzy Set Theory and Applications*, M. Gupta, R. Ragade, and R. R. Yager, Eds. New York: North-Holland, 1979, pp. 3-18.
 - [22] L. A. Zadeh, "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic," *Fuzzy Sets and Systems*, vol. 90, pp. 111-127, 1997.
 - [23] Y. Y. Yao, "Granular Computing: basic issues and possible solutions," presented at 5th Joint Conference on Information Sciences, Atlantic City, NJ, USA, 2000.
 - [24] W. Pedrycz, "Granular Computing: An Introduction," presented at IFSA world congress, Vancouver, Canada, 2001.
 - [25] A. Josang, "A logic for uncertain probabilities," *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems*, vol. 9, pp. 279-311, 2001.
 - [26] K. Stathis, O. de Bruijn, and S. Macedo, "Living Memory: Agent-based Information Management for Connected Local Communities," *Interacting with Computers*, pp. (in press), 2002.
 - [27] MPEG-7, "Overview of the MPEG-7 Standard (version 6.0)," ISO/IEC JTC1/SC29/WG11, Pattaya, Standard N4509, Dec 2001.
 - [28] F. Pereira and R. Koenen, "MPEG-7: a standard for multimedia content description," *International Journal of Image and Graphics*, vol. 1, 2001.
 - [29] R. Englemore and T. Morgan, *Blackboard systems edited by Robert Englemore, Tony Morgan*. Wokingham, England ; Reading, Mass.: Addison-Wesley, 1988.
 - [30] S. Ahuja, N. Carriero, and D. Gelernter, "Linda and Friends," *Computer*, vol. 19, pp. 26-34, 1986.
 - [31] T. J. Lehman, A. Cozzi, Y. H. Xiong, J. Gottschalk, V. Vasudevan, S. Landis, P. Davis, B. Khavar, and P. Bowman, "Hitting the distributed computing sweet spot with TSpaces," *Computer Networks-the International Journal of Computer and Telecommunications Networking*, vol. 35, pp. 457-472, 2001.

Voice Codification using Self Organizing Maps as Data Mining Tool

Juan D. Velásquez, Hiroshi Yasuda, Terumasa Aoki and Richard Weber¹
University of Tokyo, Research Center for Advanced Science and Technology
e-mails: {jvelasqu,yasuda,aoki}@mpeg.rcast.u-tokyo.ac.jp

¹*University of Chile, Faculty of Sciences Physics and Mathematics,*
Department of Industrial Engineering
e-mail: rweber@dii.uchile.cl

Abstract.

Voice transmission plays a crucial role in many applications such as e.g. telecommunications. An alternative to increase the efficiency of voice transmission is using a codification that permits compressing the signal to be transmitted. Such a compression assumes data sets with basic forms, whose combination produce the voice signal. Generally this data set is organized around an array of data, known as codebook. The codebook is constructed by a vectorial quantization process, which consists of looking for what vectors are most representatives, within a set. Next a structure of data is created that stores the vectors, also known as centers. Then, given a codebook with the most representative basic forms, the problem is translated to take a piece of voice, look for its position and transmit it. Since the receiver will have the same structure of data the voice will be able to be synthesized. The problem consists in the search in the codebook, which can be expensive in terms of computation and other resources, which perform operation in real time, characteristic that in some services is fundamental, for example in telecommunication. In this work we present a new algorithm to construct and to cross codebooks by using a data mining tool such as self organizing maps over a database of humans voices. This algorithm produces a codebook structure within a relation of proximity between its elements, reducing the problem to a local search, which allows to decrease compression time and to reduce the rate of transmitted bits.

1 Introduction

In order to be able to transmit the digitized human voice, it is necessary to take samples from the signal with certain frequency [3, 8], that in the traditional case, corresponds to 8,000 per second. Each sample is codified using 8 bits. Consequently, if it is desired to transmit a second of voice, a channel of $64.000 \frac{\text{bits}}{\text{sec}}$ will be necessary. In some cases, the exposed number is critical and produces congestion in the services that are desired to implement, e.g. in the case of wireless telephony.

Several techniques have been developed in order to reduce the amount of bits to be transmitted by a channel, which contemplate alternative methods of codification and digital compression of the voice.

One of the mostly used techniques is the creation of structures of data that allow to store pieces of the signal, with the purpose of creating a vectorial quantization, i.e. a symbol represents a vector of samples. The structures that support this functionality, are known as codebook.

Such a compression is based on the existence of a finite set of basics forms (signal segments) whose combination creates the voice signal. Then, given a large array or codebook with all the possible basics forms, the problem is to take a piece of voice, to search for its position of the array and to transmit it. Since the receiver will have the same data structure he/she will be able to be synthesized the voice.

The array of codebooks could be of a very high dimensions. Nevertheless, when analyzing a graphical intensity versus time of the voice signal, it is possible to observe that many patterns are very similar in form, which results in a finite array size which will only have to contain those patterns that are the base of others. The problem now is to know which are these basic patterns and the necessary number in order to make a reasonable compression, with respect to the quality of the synthesized voice.

One of the mostly used techniques in the codification of the voice signal **Lineal Predictive Coding** (LPC)[1], stores the intrinsic properties of a signal segment, the surrounding of the spectrum and allows a more compact description.

The codebook [4, 9] is created from the LPC synthesis, using previously recorded human voices within a high phonetic content. Next, techniques are applied in order to obtain the most representative LPC of a set, creating the codebook [10].

Generally the codebooks have 1024 LPC. In order to find the most representative LPC for a piece of voice a complete search in the codebook is necessary, since there does not exist an order relation among LPC.

In this work we present a new form to construct codebooks and to make a less expensive search. Based on the assumption that the signal of synthesized voice is continuous, it will only be necessary to review part of the array, settling down a local search.

In order to obtain continuity, the codebooks will have to be created in a different way compared to the traditional one [4], i.e. the selected LPC will have to fulfill characteristics that show proximity to each other.

In section 2, we explain how LPC from the voice signal are synthesized. Section 3 shows the measurement of used distortion to compare two LPC. Section 4 explains how a self organizing map was used to create codebooks from the synthesized LPC, taking as a basis voices of different speakers. Section 5 shows how the newly proposed search is performed in a codebook. Section 6 presents the application of the suggested technique for compression of a database of human voices. Section 7 compares the results derived by our technique with results obtained by traditional methods showing the advantages of our approach. Finally, section 8 shows the main conclusions of the present work.

2 Lineal Predictive Coding

The estimation of the parameters that compose a LPC, is based on the fact that a prediction of parameters can be performed for a certain data set previous to its appearance.

A signal s_n can be defined as the output of a system [1] with a certain input u_n . The equation 1 shows the prediction of the signal s_n , where a_k and the gain G are parameters of the system.

$$s_n = \sum_{k=1}^p a_k s_{n-k} + G u_n \quad (1)$$

For a signal s_n , the problem consists in determining the coefficients of prediction a_k and the gain G , which is reduced to calculate the elements of the filter which minimizes the mean square error between the real signal and the considered one.

Be \hat{s}_n , the considered signal and we suppose $u_n = 0$. The prediction error is given by the equation 2.

$$e_n = s_n - \hat{s}_n = s_n - \sum_{k=1}^p a_k s_{n-k} \quad (2)$$

In the least square method the parameters a_k are obtained minimizing the squared error with respect to each parameter.

$$E^2(e) = \frac{1}{2} \|e\|^2 = \frac{1}{2} \sum_{n=0}^N (s_n - \hat{s}_n)^2 = \sum_{n=0}^N \left(s_n - \sum_{k=1}^p a_k s_{n-k} \right)^2 \quad (3)$$

In order to find the parameters a_k that minimize the error, the equation 3, has to be derived with respect to each variable and each derivative should be set to zero.

$$\frac{\partial E}{\partial a_i} = 0 \quad 1 \leq i \leq p \quad (4)$$

$$\frac{\partial E}{\partial a_i} = \sum_{n=0}^N (s_n - \sum_{k=1}^p a_k s_{n-k}) s_{n-i} = 0 \quad (5)$$

Thus the system of equations is obtained 6.

$$\sum_{k=1}^p a_k \sum_{n=0}^N s_{n-k} s_{n-i} = \sum_{n=0}^N s_n s_{n-i} \quad 1 \leq i \leq p \quad (6)$$

The autocorrelation is defined as:

$$R(i) = \sum_{n=-\infty}^{+\infty} s_n s_{n+i} \quad (7)$$

which is an even function, that is to say, $R(i) = R(-i)$.

The autocorrelations are calculated on finite sections of the signal. If we suppose that the signal is null before and after these sections, the system of equations 6 can be written as

$$\sum_{k=1}^p a_k R(i - k) = R(i) \quad 1 \leq i \leq p \quad (8)$$

By the previous assumptions, the present system in the equation 8 has a symmetrical matrix of Toeplitz type (in a Toeplitz matrix the diagonals elements are equal). This allows that the system can be solved by the recursive method of Durbin [4, 9], that determines the parameters in a reduced number of operations.

$$\begin{bmatrix} R_0 & R_1 & R_2 & \dots & R_{p-1} \\ R_1 & R_0 & R_1 & \dots & R_{p-2} \\ R_2 & R_1 & R_0 & \dots & R_{p-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ R_{p-1} & R_{p-2} & R_{p-3} & \dots & R_0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \\ \vdots \\ R_p \end{bmatrix} \quad (9)$$

The equations are recursively solved for $i = 1.2...p$, being the final solution:

$$a_j = a_j^{(p)} \quad 1 \leq j \leq p \quad (10)$$

3 Comparison between two LPC

Although the LPC is a vector of p elements, two such vectors cannot be compared through traditional distances like e.g. the euclidian one. For example, the figure 1 shows two LPC that have common characteristics in their components. If the euclidian distance is used, apparently they are very similar, but they can synthesize totally different signals.

LPC 1									
1.5	0.5	0.1	1	0.4	0.9	0.7	1.2	0.1	0

LPC 2									
1.5	0.5	0.1	1	0.4	0.9	0.7	1.2	0.1	1

Figure 1: Comparing LPCs

For this reason, another "distance" has been developed, that in fact is a measure of distortion, like the one of Itakura-Saito [7], that not necessarily fulfills the triangular inequality.

This measure of distortion is used to obtain a notion of how different two LPCs are and is described in the equation 11.

$$d(\vec{X}_1, \vec{X}_2) = (\vec{X}_1 - \vec{X}_2)^T \cdot R_{\vec{X}_1} \cdot (\vec{X}_1 - \vec{X}_2). \quad (11)$$

where :

- \vec{X}_1 : LPC Vector with which LPC \vec{X}_2 . is compared
- \vec{X}_2 : LPC Vector of the signal of voice to consider
- $R_{\vec{X}_1}$: Matrix of associated autocorrelation of \vec{X}_1 . This matrix contains the autocorrelation coefficients generated for $LPC_{\vec{X}_1}$.

This measure of distortion has proven to give better results than the traditional ones, but is not the only one. There is a lot of current research focussing on the development of metrics that allow to measure how similar two different LPCs are.

4 Self Organizing Map in the creation of the codebook

The used map, is an artificial neural network of Kohonen type [2, 5]. Schematically, it is presented as a two-dimensional array in whose positions the neurons are located. Each neuron is constituted by a n -dimensional vector, whose components are the synaptic weights. By construction, all the neurons receive the same input at a given moment.

The idea in this learning process is to present an example to the network and by using a metric, to search the neuron in the network most similar to the example (center of excitation). Next we have to modify its weights and those of the center's neighbors.

This type of learning is called **not supervised**, so that the neurons "will move" towards the centers of the groups of examples that they try to represent. The previous process was proposed by Kohonen, in his *Algorithm of training of Self Organizing networks* [5].

A very important characteristic of this type of network, is that it can make a vectorial quantization, i.e. [4], from a set of n elements, it can find m ($n \gg m$) more representative ones and associate an indicator to them. With the distance and suitable training, it is possible to use the network of Kohonen to construct codebooks of LPC. The fundamental difference with an algorithm of similar purpose [10], is that a certain degree of topological continuity between the centrioles is obtained, giving the codebook a notion of neighborhood between its elements.

4.1 Operation of the Map

The notion of vicinity among the neurons gives diverse topologies. In this case the toroidal was used, which means that the next neighbors of the neurons of the superior edge, are in the inferior and lateral edges (To see figure 2)

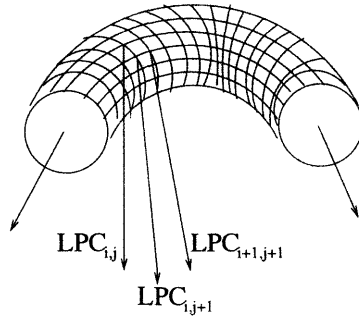


Figure 2: Proximity of the LPC in a network of toroidal Kohonen

With this topology, and using the Itakura-Saito distance, we are able to maintain locality in the search of the LPC. This means, when looking for a LPC instead of revising all codebook in order to find the following LPC we only have to review a sector, since there is a high probability of finding it in a next vicinity (to see figure 3).

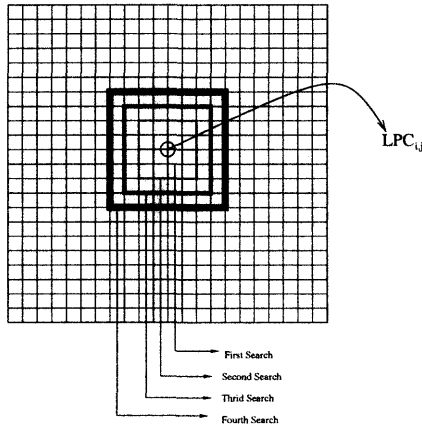


Figure 3: Search in codebook

The search mentioned before is an approximation, since it is possible that the LPC that best represents the piece of voice to be compressed, is not close to the selected one.

Table 1 shows which is the probability of finding a LPC near its successor. Define as *Radio of the Square R* as measured of vicinity between LPCs. We will say that a LPC is near in R when $2R$ is within a square of sides with respect to another LPC.

With the previous results, the search is made according to the following algorithm:

1. Be LPC_{i-1} the last one chosen and LPC_i the one that will be chosen of the codebook.
2. $LPC_i = LPC_{i-1}$, be R the radius of the square, $R=1$.

Radio of the Square R	% LPC neighbors to chosen
0	55.754
1	58.914
2	62.513
3	75.629
4	80.676
5	83.112
6	85.233
7	90.997
8	91.676
9	95.884
10	96.349
11	97.887
12	98.976
13	99.112
14	99.887
15	100.000

Table 1: Statistic of proximity between LPC

3. To display the LPCs synthesized by the system to which they are within the square of radio R . Is LPC_R the chosen one.
4. If LPC_R it is better than LPC_i , then, $LPC_i = LPC_R$, $R = R + 1$ and to return to point 3. In the opposite case, stop the search and LPC_R is the chosen one.

Fig. 3 contains the operation of the search. The most outstanding squares, show the progress of the search, which can cover all codebook, in the worst case.

5 Application

The developed technique was used in a digital compressing system of voice, specifically in the codification module where a vectorial quantization of the voice signal is made.

The effectiveness of the module was proven the test MOS, which measures the quality of synthesized voice.

The construction of the database with the human voices, was created remembering a high phonetic variability and the ages of the speakers.

5.1 Selection of the phrases for training

In the creation of the training base five types of phrases were considered:

- Affirmations: Most of the phrases that take care daily are of this style and they are characterized to have declinations with little variation of the tone.

- Questions: They are characterized, mainly, by the fact that at the end of the phrases, increases the tone of the declination, accentuating itself the presence of the last syllables.
- Exclamations: They mainly have marked the syllables of the beginning and end of the phrase, being its declinations similar to those of the affirmations.
- Aggressive declination: Although their declinations are similar to those of the orders, they are characterized to have fricatives or affricative syllables, and/or excessively accentuating those words already accentuated. They are the classic phrases that show the anger emotion.
- Passive declination: It is not a formal declination of the cultured language. It is characterized by the extension of some syllables within the phrase, generally of whispering way. Sometimes it is used in substitution to a formal question

5.2 Conformation of the Group Shows

The two main variables that were used in the recording of the phrases, were sex and age. This is due to the fact that women and men have different characteristics of their voice (base frequency), being for women and children, generally, greater than for men. It was considered fundamental, in order not to create slant, that the number of men and women was similar. Altogether 1150 phrases from people of different ages were recorded, being 12 to 60 years old.

5.3 Test of quality

The MOS (Mean Opinion Score) consists of a test of subjective quality on the voice synthesized by a system. Basically a series of recordings appears to a group of people that after being listened, are evaluated in a scale from 1 to 5 (1=bad, 2=regulate 3= acceptable, 4=good and 5= excellent).

The score obtained by the system is 3.4, entering the category of "acceptable" according to test MOS. It is slightly smaller than the standard required for most of the wireless systems of telecommunications.

5.4 Comparison with another method

The LGB algorithm [10] is traditionally used to construct a codebook. It used the Euclidean measure in order to find the most representatives LPC in a voice data base. Next, it is possible to create a codebook using the result of the LGB algorithm. However, this techniques did not continue among LPC, because the measure used does not consider the special characteristics of the data type. Additionally, the construction of the codebook does not use proximity among the border cases.

Table 2 shows the proximity among LPCs using the LGB algorithms. Comparing this result with table 1 reveals the difference in the searching procedure. Using the LGB algorithm, in 60% of the cases we have to search in 196 LPC. Using the algorithm proposed in this work, it only requires a search in 16 LPC.

Radio of the Square R	% LPC neighbors to chosen
0	10.324
1	12.432
2	18.134
3	24.793
4	35.672
5	41.224
6	50.374
7	61.448
8	69.882
9	78.446
10	87.678
11	93.787
12	95.743
13	98.358
14	99.476
15	100.000

Table 2: Statistic of proximity among LPC using LGB Algorithms

Additionally, the proposed method has two important improvements at the level of the quantization module:

- It increases the speed in the quantization by 50%.
- The amount of necessary bits for codification is reduced by approximately 30%.

6 Conclusions

The search in the codebook of 1024 LPCs can be reduced by 85%, i.e. only 144 LPCs have to be considered, which in execution represents approximately 86% less than in the case of exhaustive search.

The continuity in the codebook, allows that the differences between the indices of the LPC have little variability, which can be used in order to reduce even more the rate of sent bits [6] to transmit the index.

The Kohonen neural network can be used as a mechanism for codebook generation, as long as the measurement of distortion allows to quantify the differences between two LPC.

This method can be used in other areas of data processing. As future work, it is proposed to use the topology of toroidal networks to analyze behavior of customers. In particular we want to study variables that influence in the process of purchase decisions. In this case, the vector that appears to the network, is a compound of characteristics relative to the habits of purchase and personal data.

References

- [1] J. Makhoul, Linear prediction: A tutorial review, *Proceedings of IEEE* 1975 vol. 63, pages 561-580.

- [2] Ben J.A. Krose and P. Patrick Vander der Smagt, An introduction to neural networks, *University of Amsterdam* 1994.
- [3] A. S. Spanias, Speech coding: A tutorial review, *Proceedings of the IEEE*, vol. 82, no. 10, Oct. 1994.
- [4] Robert M. Gray, and Allen Gersho, Vector Quantization and Signal Compression, *Kluwer Academic Publishers* 1992 pages 127-128.
- [5] Teuvo Kohonen, Self-Organization and Associative Memory, *Springer-Verlag* 1987, 2nd edition.
- [6] James L. Massey, Applied Digital Information Theory, *ETH Zurich* 1991, chapter 2.
- [7] Rabiner L. and Juang B-H, Fundamentals of Speech Recognition, *Prentice Hall, Englewood Cliffs, New Jersey* 1993
- [8] Alan V. Oppenheim and Ronald W. Schaffer, Discrete-Time Signal Processing, *Prentice Hall Signal Processing Series* 1999
- [9] Juan Velásquez, Aplicaciones Avanzadas de Redes Neuronales al Desarrollo y Simulación de un Sistema Compresor Digital de Voz, Memoria de Título, *Universidad de Chile, Facultad de Cs. Físicas y Matemáticas, Departamento de Ingeniería Eléctrica* 1996
- [10] Y. Linde, A. Buzo and R. Gray, An algorithm for vector quantizer design, *IEEE Transactions on Communications*, Vol. 28, pages 84-95 1980.

Identifying Patterns of Corporate Tax Payment

Maria Cleci MARTINS
Business School
University of Gloucestershire
Cheltenham, UK
mmartins@glos.ac.uk

Iria M GARAFFA
Business School
Brazilian Lutheran University
Canoas, Brazil
iriag@terra.com.br

Abstract

This research investigates the possibility of identifying patterns for corporate tax payment behaviour behind the Government database on companies. The aim is to reveal companies showing abnormal behaviour or outliers (companies whose behaviour on tax payment is away from the average). Knowing whether or not the outliers are evading taxes will be the result of in site inspections which are under development. We are particularly interested in analysing whether there is correlation between the dependent – the tax paid in earlier years – and the independent variables, such as total revenue and number of employees. A preliminary data analysis shows that there is strong correlation between those variables, and therefore the data might be appropriate for the task. However, patterns are only unveiled after using a two-step methodology: Clustering and Backpropagation Neural Networks (BNN). Clusters of similar companies are investigated separately using BNN. Analysis of results from the Artificial Neural Networks (ANN) was based on the comparison between the forecast and the true amount of tax paid by companies each year.

1 INTRODUCTION

Tax evasion is a serious problem for governments. Usually the common citizen is the taxpayer having no alternative but to pay taxes. Companies receive those taxes, which are added to the price of the product or service, and pass them on to the Government. It has always been a concern that some companies might be keeping that money using artificial accountancy means and therefore increasing tax evasion. Thus, they would be cashing on money that otherwise could be employed by governments to benefit taxpayers.

In Brazil, companies are very careful in their accountancy to avoid as much as possible transferring those resources to the government. A legal mean for that is supporting tax deductible activities such as Art, Cinema, Museums and charities.

In spite of those legal activities, companies may also try and avoid taxes payment by manipulating their accountancy. Tax officers usually do not have many tools to discover the true tax amount that companies should have paid. The most used approach is sending Officials to perform in site inspections on the company documents and redoing calculations. However effective an inspection can be, there is always the possibility for the company to hide important information.

Therefore, the tax system relies on in site inspection of accountancy notes and books to investigate whether a company is not breaking the tax law. However, the resources

available are not enough to develop a full investigation in all companies every year. Therefore, a small sample of companies is thoroughly analysed each calendar year.

The decision process on what companies are scrutinised each year is based on overall performance of the company regarding some criteria, such as companies that have not been investigated in the last 10 years. Depending on the inspection timetable, some companies might never be inspected. It is worth emphasising that, in Brazil, due taxes referring to transactions older than 5 years are redeemed and do not need to be paid. Thus, to reduce tax evasion, the gap between inspections should be not longer than 5 years. If authorities could use a systematic approach based on company data over the years it would be possible to detect whether a recently inspected company still shows abnormal behaviour in relation to its peers.

Governments hold a massive database about companies' activities that might be useful to identify patterns of tax payment behaviour depending on variables such as revenue and number of employees. We have investigated a sample of such database containing anonymous information about companies in a particular region of Brazil. A preliminary analysis of the database showed that there is strong correlation between the dependent and the independent variables. However, attempts to train a Backpropagation type of Neural Network (NN) to detect patterns underneath the data were unsuccessful. Looking further into the correlation between variables we have found that the high rate was heavily influenced by some companies larger figures. Segmenting the data using the unsupervised clustering allowed a closer analysis into the patterns behind each group of companies. This is explained by the fact that companies belong to different tax regulations depending on the type of activity they are in, the location of their plants, etc.

This research investigates the possibility of finding patterns of behaviour regarding tax payments that would allow governments to develop a more effective in site inspection timetable. Although we do not know whether an outlier is evading tax, the methodology gives an indication of which companies should be inspected more often.

This paper is organised as follows: Section two roughly explains the Tax System in Brazil, and the basics of Clustering and Neural Networks techniques. The data and the methodology used are explained in Sections 3 and 4 respectively. Section 5 shows the experiments and results, followed by a conclusion in Section 6.

2 THEORETICAL BACKGROUND

2.1 TAX SYSTEM IN BRAZIL

According to Brazilian law, every commercial transaction must be identified in business documents. The information should then be sent to official authorities and stored in databases which are used for statistics, planning, etc. We have been investigating part of such database so as to identify possible corporate tax payment patterns. Besides the relationship between dependent and independent variables, we could also analyse data consistency over the years. That means we were looking for temporal pattern regularity inside the data. Therefore, wherever the data, if it is inconsistent from one year to another, the methodology should be able to detect it. This result would give authorities some indication about which companies to inspect first.

ICMS is a type of tax paid on sales transaction of goods or services to the end customer. Its value represents in average 17% of the value of the product or service. There is specific legislation which is applied to companies, depending on their activity (i.e. industry, commerce, import), nature of the product (for instance, food products are lower taxed).

There are no official figures about the amount of tax evaded in ICMS every year in Brazil. According to Sintaf¹, in 2001, only one Brazilian county has received around US\$ 2.6 billion in ICMS type of tax, which represents 7% of the amount of ICMS tax paid by taxpayers in the whole country. There are estimates that approximately 30% of the Brazilian economy is not registered (also known as informal). Most of those transactions involve products and services belonging to registered companies that do not accomplish with legislation regarding accountancy registration. That gives one an idea about the likely amount of tax being evaded each year in the country and the importance of finding effective ways of detecting it.

2.2 ARTIFICIAL NEURAL NETWORKS

Artificial neural networks (NN) are a family of parallel computing devices. Within this family there is considerable diversity, but all neural networks share some basic common features. They are composed of a number of simple processing elements known as neurons. These neurons take data (X_n) in from a number of sources and compute an output dependent on the values of the inputs using an internal transfer function (Figure 1).

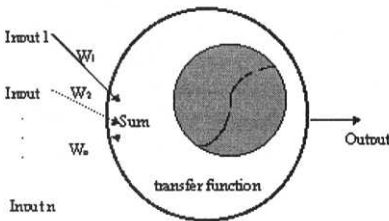


Figure 1 – Representation of a Neuron

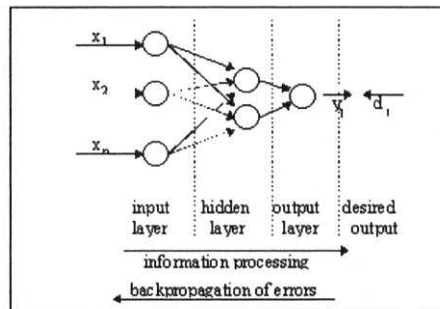


Figure 2- Backpropagation NN

Neurons are joined together by connections; data flows along these connections and is scaled during transmission according to the values of weights associated with the connections. The output of a neuron may therefore contribute to the input received by another. Some connections take data in from an external source whilst others pass data back out.

The neural network's functionality is bound up in the values of the connections weights (W), which can be updated, causing the network to "learn". Learning is a procedure that allows the network to adjust itself by means of gradually changing the network connection weights. A neural learning procedure can be *supervised* if the data used to train the network contains a desired output; otherwise *unsupervised* training takes place.

Backpropagation is a particular type of artificial neural network and is structured as shown in Figure 2. Salient features include the following:

- i. The network has no circular information paths but is of a feedforward nature. Information flows from left to right in the diagram shown. Optimization is performed in the reverse direction using the desired output as the NN target output.
- ii. The neurons are arranged in layers. The output from each neuron in a given layer is connected to the input of every neuron in the next layer along.

¹ Tax Officials Inspectors Union (Sindicato dos Auditores Fiscais) - A economia por baixo do pano – Jornal da tarde –available at <http://www.jt.estadao.com.br/editorias/2002/06/02/eco017.html>

- iii. The size of the input and output layers (Y) is defined by the size of the input (X) and desired output vectors (D), but the "hidden layer" has no such constraint and can contain an arbitrary number of neurons. It is possible to have more than one hidden layer.
- iv. Neurons in the input layer have one input each and therefore have no real functionality. The notion of an input layer is therefore purely for the purposes of abstraction.
- v. The neurons contain a smoothed step function (such as a sigmoid) as a transfer function.

Training takes place using a gradient descent algorithm[2], which distributes the global error (difference between the network current output and the desired output) over the various neurons as a "local error". Weights are updated according to the size and direction of negative gradient on the error surface. The hope is that over time, the desired and actual outputs converge. The network has then 'learnt' to represent the function, which relates input to output.

In fact a Backpropagation network can theoretically represent any arbitrary function with at most two hidden layers [3], hence its attractiveness. However, this theoretical result does not specify any limit on the size of data set required, the size of the hidden layers needed and the number of iterations required to train the network. Given that the data set is always finite in practice, this places some limitations on the technique. In particular, one encounters the problem of over training, which exhibits itself by a good fit to the training data, but accompanied by poor powers of generalisation to unseen data items. This can occur either if the hidden layer(s) is too large and/or if the network is trained for too many iterations; in such circumstances the neural network tends to learn the individual data items rather than generalising. Thus it is important to quantify the effectiveness of the technique *in an operational context* and the optimum configuration has to be determined by experimentation. Furthermore, comparative testing must be carried out using an independent test set, which is not used for training. This description is necessarily brief but the paradigm is well described in textbooks[4].

2.3 CLUSTERING

Clustering is a methodology to group individuals (in our case, companies) according to their similarities. The idea is to find a criterion in which elements of a cluster are homogeneous and elements of different clusters are dissimilar in terms of the "m" attributes. Known clustering algorithms are K-means, K-nearest neighbour, etc. [5]). The performance analysis can be developed using probabilistic tool [6]). Learning is the key point for clustering. It drives the way the data is analysed or investigated. Knowing the class beforehand means that the observations need only to be classified according to the known classes. Thus, a priori knowledge about the categories (or clusters) in the data suggests the usage of supervised learning methods to classify observations. However, there are many situations where the classes are not known. For instance, customers can be clustered according to their shopping behaviour. The lack of information about the type and number of classes suggests the adoption of unsupervised learning methods. This unsupervised clustering problem is highly non-linear and results are found given limitations of number of iterations, number of clusters, etc. When the cohesion of a cluster is high, i.e., the examples in it are very similar, it defines a new class.

2.3.1 Unsupervised clustering

The statistical approach for clustering is based on assumptions about the random process that generates elements as points in attribute space. The probability density for each element is the mixture of densities for each cluster,

$$\sum \pi_k P_k(., \theta_k)$$

π_k probability of the cluster k occur

θ_k parameters vector characterising the cluster k

P_k probability density for cluster k

The likelihood function for the data x_1, \dots, x_n is the probability density for the entire data set:

$$\prod_{i=1}^n \sum_{k=1}^p \pi_k p_k(X_i; \theta_k)$$

The estimates of parameters π_k and θ_k to assign element i to cluster k are obtained by maximising the likelihood function for which $p_k(X_i; \theta_k)$ is the largest.

Maximum likelihood approach cannot be used to find the optimum number of clusters, as the maximum value increases as the number of clusters increases [7]. Bayesian statistical analysis can be used for that task, provided that the prior distribution for the number of clusters and for all of the π_k and θ_k parameters is known.

AutoClass² [8] is public domain software for unsupervised Bayesian classification that seeks a maximum posterior probability classification. That means that it finds the most likely set of classes with respect to the data, the probabilistic model and the number of iterations allowed. It accepts many types of variable formats either qualitative or quantitative, as well as missing values.

3 DATABASE

The data was composed of 12710 cases (around 3000 companies) referring to companies' activities developed during the period 1996-1999. The data reported value for the following variables:

- i. Year that the data was collected – (ranging from 1996 to 2000)
- ii. Year of the company's foundation
- iii. Total revenue (in Brazilian currency)
- iv. Physical area used by the company (in squared meters)
- v. Number of employees (registered with the company)
- vi. Electricity used during the year (in kilowatts per hour)
- vii. Taxable Revenue (reports the value of all goods the company bought in the period which were subject to taxes)
- viii. Value of the company's supplies which taxes have to be deducted
- ix. Taxes paid by the company to the government (this is the dependent variable)

Monetary values are expressed in Brazilian currency (i.e. Real). No deductions to account for inflation were made. The Brazilian economy has been stable since 1994, and we did not assume that the lower rate of inflation would be troublesome for the forecasting.

Number of employees refers only to those registered with the company. Brazilian informal economy is significant, but there is no official information about it. Estimates are that

² <http://ic-www.arc.nasa.gov/ic/projects/bayes-group/autoclass/>.

informal employment rate is approximately 25% of the active population³. However, because it is informal (and sometimes illegal), companies and individuals in this part of the economy do not wish to fill forms about their activities.

Taxable revenue reports monetary values that enter the company (including taxes paid by individuals) mostly through sales. This value is important, as it is the base for the company to find out how much tax should be paid to the government.

Supplies bought – In Brazil, companies can deduct the taxes paid to their suppliers from the taxes they have to pay to the government (i.e. they pay the difference between the tax on taxable revenue and the tax they have already paid through their suppliers).

Tax paid – a curiosity in this variable is that it can be negative, meaning that the company in that year had more tax to deduct than to pay. It might happen that the company, in that year, had bought more raw and processing material than its sales. There are also other reasons for such behaviour that is not in the scope of this research.

3.1 PRE-PROCESSING THE DATABASE

The database had both quantitative (5 variables) and qualitative data (2 variables). Example of qualitative data was the year the company started activities (foundation). Qualitative variables were then transformed into quantitative so that we could use a quantitative tool such as Neural Network. We were also interested in using the temporal information without employing a more specific tool (such as time series analysis). We decided that we would use an additional variable that would report the time between the year of the company's foundation and the year the tax was paid. Thus, after transformed, we have 9 (nine) variables (8 dependent and 1 independent).

The data showed good correlation among dependent and independent variables (see Table 1). Strong correlation was between tax paid and variables taxable revenue, value of supplies bought, electricity usage, and number of employees. Although correlation seemed high, attempts to employ forecasting methods in this data failed. Investigated Neural Networks showed no learning. We decided to analyse correlation for segmented groups of companies, according to the tax paid. Then we discovered that the correlation was low (not higher than 10%) for about 70% of the companies. On the other hand, inverse figures were found for big companies. This is explained by the fact that larger companies do tend to record their transactions more accurately than smaller companies. Another reason for that is that smaller companies eventually benefit from tax exemption and other law regulations and therefore the relationship between variables tend to be modified. However, we did not know precisely which companies were similar. Therefore, we decide to use clustering technique to groups companies.

Table 1 - Data Statistics and Correlation between Variables

Code	Variable*	Mean	Std Dev	Minimum	Maximum	Correlation to tax paid
1	AGE	11	9	1	86	0.0811
2	AGE_TAX	9	9	1	83	0.0845
3	AREA	800	8700	1	481296	0.1356
4	EMPLOYEE	13	41	1	985	0.8523
5	ELECTRIC	78583	411690	1	11823330	0.8079
6	TAXABLE REVENUE (R\$)	1545565	7900611	35	318000000	0.7213
7	TOTAL REVENUE(R\$)	1472357	14044171	1	1330000000	0.4076
8	VALUE OF SUPPLIES BOUGHT (R\$)	1249287	6970981	21	317000000	0.6195
9	TAX_PAID (R\$)	296374	1380153	-5247868	34573441	1

³ <http://www.worldpaper.com/Archivewp/1999/sept99/castilho.html>

4 METHODOLOGY

Neural Networks training is comparable to conventional calibration procedure. However, trained Neural Network coefficients are difficult to analyse and draw conclusions about the quality of the results. Therefore, goodness-of-fit of a trained network is evaluated against a testing file that was not used in the training phase. To obtain a testing file, we decided to randomly split the data into 90% for training and 10 % for testing. We are aware that by doing so some important information might be left out in the training procedure, causing the net to produce poorer results. Other approaches that could be used as mean of comparison are to split the data in different ways, training the network and then finding a mean acceptable performance. However, these approaches are also attempts of finding better results.

4.1 NN TRAINING AND TESTING

In this research, we are interested in forecasting the tax that companies should have paid, given the information about their activities. The NN training is based on setting NN parameters for different configurations. The number of input and output nodes must accord with independent and dependent variables in the data, respectively. Unlike Regression techniques, NN training allows one to specify as many hidden neurons as the data requires accounting for hidden nonlinearities. Analysis from earlier experiences [1] showed that NN with either two hidden layers, or many neurons in a hidden layer could be easily overtrained. This implies good performance replicating the training data, but not extrapolation. However, we did try to train BNN with two hidden layers and results were poorer.

First NN training attempts using the whole data showed no learning despite the number of configurations and types of NN tried (such as Radial Basis Function and Backpropagation type of NN). Only after segmenting the data into groups of companies with similar patterns that NN showed good learning rate within each cluster.

Analyses of the results were performed for each cluster using a testing file containing approximately 10 % of the data belonging to the cluster. Three types of measures were investigated:

Correlation among dependent and independent variables, and its standard deviation.

Absolute difference (AD) between the tax-due forecasted by the network and the tax effectively paid by the company, and the standard deviation of AD.

Relative difference (RF) between the error and the tax paid by the company.

The performance measure used to evaluate the quality of the results reveals the strategy the Government would adopt to combat tax evasion. If the strategy were to reduce tax evasion in value, the best performance measure would be to investigate the absolute difference. However, if the objective was to find out companies' irregular behaviour in this matter, then the relative difference should be used.

5 EXPERIMENTS

5.1 CLUSTERING

There was not possible to know beforehand the class a company would belong to only by looking at the values of the variables characterising it. Therefore, we used unsupervised clustering methodology to segment the data. Autoclass software was employed and the Normal probabilistic model was assumed as a rule for clusters formation. Table 2 reports mean values of the variables (dependent and independent) in each cluster of similar companies (numbered from 0 to 6). Clusters showed were found using Autoclass, assuming a Gaussian model.

Table 2- Clusters of companies

Cluster	Age	Age_tax	Total revenue(R\$)	Area	Emp.	Electricity	Value supplies bought(R\$)	Taxable revenue(R\$)	Tax paid (R\$)
0	13	11	552611	463	6	25678	574210	477238	97542
1	11	9	90468	182	2	7935	91643	72468	19176
2	7	5	27288	84	1	2637	27952	21402	6550
3	12	10	215439	245	3	12390	220350	180222	40126
4	15	13	1711278	774	18	75425	1822704	1498244	324391
5	16	14	11117199	5193	96	668853	12930136	10380939	2547903
6	7.7	6	1795300	105	3	19711	163647	142254	21446
Mean	12	9	1565364	842	14	83433	1642527	1327278	315227

An analysis of the correlation between the dependent and independent variables within each cluster shows the differences between clusters (see Table 3). Larger companies are in clusters 4 and 5, which also hold high correlation rates between the dependent and independent variables. For instance, in case of cluster “0”, the variable Age (means age of the company) holds a weak correlation (-0.022) with the dependent variable (tax paid). On the other hand, Total Revenue showed a correlation of 0.432.

Table 3- Correlation dependent x independent variable in each cluster

Variable	Cluster 0	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
AGE	-0.022	0.017	0.080	-0.012	-0.003	-0.063	0.141
AGE_TAX	-0.016	0.032	0.095	0.005	0.000	-0.065	0.155
AREA	-0.010	0.038	0.057	0.006	-0.012	0.077	0.020
EMPLOYEE	0.068	0.151	*	0.118	0.358	0.805	0.083
ELECTRIC	0.070	0.115	0.062	0.074	0.138	0.783	0.010
TAXABLE REVENUE	0.097	-0.193	0.138	-0.181	0.350	0.525	0.299
VALUE OF SUPPLIES BOUGHT	0.440	0.437	0.550	0.416	0.574	0.641	0.355
TOTAL REVENUE	0.432	0.396	0.424	0.403	0.560	0.692	0.003
Data cases	2151	1985	2074	1943	1539	1100	815

* Companies in this group had only one employee

5.2 FORECASTING

Each clustered data (clusters “0” to “6”) had to have a complete set of NN training and testing procedure. Therefore, we developed approximately 40 BNN training and testing sessions. Several BNN configurations were used in order to find the best one for each cluster. However, only results obtained for companies belonging to one of the clusters are shown next (i.e. cluster “0”).

Table 4 shows forecasting results from Backpropagation Neural Network training attempts to cluster “0”. Best results according the before mentioned performance measures are highlighted. For cluster “0” we have trained and tested 5 configurations, with different number of training iterations. The best results were produced by the network with 8 hidden neurons and trained 15 thousand times.

The testing data not used in the training phase for the shown cluster was composed of 152 companies. The trained network produced results (forecast tax that the company should had paid) showing only a small difference between the forecast and actual output. That means that those companies might have paid the correct amount of tax due. However, 21 companies showed a very bad performance (difference between the tax paid and the one forecast by the network). As we said at the beginning of the paper, we are now developing

the necessary inspections in those outliers companies to verify whether the abnormal behaviour is an indication of tax evasion.

Table 4 - NN forecasting results for Cluster "0"

CORRELATION AND STANDARD DEVIATION					
SERIES	881- 5 K *	881- 10 K	881- 15 K	8421-5 K	8421-10 K
Correlation	0.9731	0.9969	0.9973	0.9730	0.9969
Std deviation	14567	4993	4604	14601	4943

ABSOLUTE DIFFERENCES					
Mean	19239	4601	3187	16350	5224
Std deviation	15499	4529	4275	13119	5568

RELATIVE DIFFERENCES					
Forecasting error %	0.3447	0.1896	0.1590	0.4645	0.2910

*K = thousands iterations
881 = 8 input neurons, 8 hidden neurons, and 1 output neuron

Figure 3 shows the comparison of the results obtained from NN and the actual tax paid by companies. It can be seen that the trained BNN could capture the data behaviour for most of the companies belonging to this group. Results for other clusters were also in a similar fashion. For instance Figure 4 pictures forecasting results for companies in cluster 4.

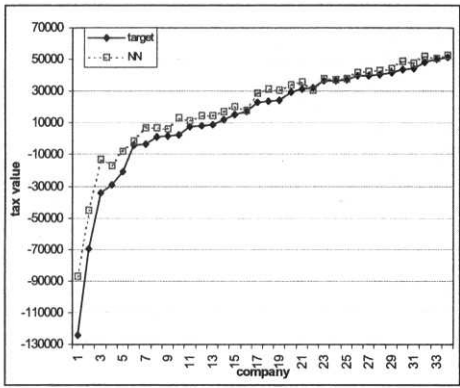


Figure 3 - Comparison for Cluster "0"

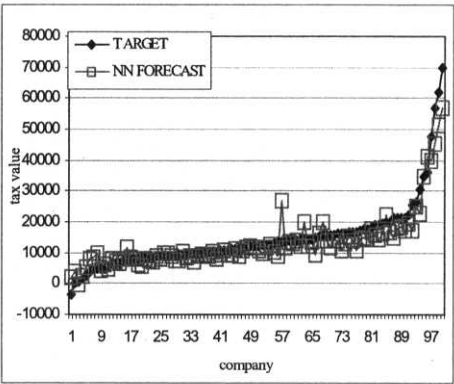


Figure 4 – Comparison for Cluster "4"

6 CONCLUSION

This research aimed at finding a methodology to investigate the Brazilian Government database on companies' activities with the purpose of discovering corporate payment tax patterns. Initial attempts to train the data using Neural Network methodology were unsuccessful. Trained networks did not show any acceptable patterns learning performance, no matter the configuration or type of net tried.

An analysis of the correlation showed that the high correlation between the dependent and independent variables were due to some companies' high figures. In order to consider the different types of tax regulations according to the company size and economic activity in Brazil, we divided the data into cluster of companies showing similar data behaviour. Groups found using unsupervised clustering approach still showed correlation between

variables. Afterwards, Backpropagation type of Neural Network was employed to each cluster to uncover the specific cluster pattern behaviour for corporate tax payment.

The two step methodology consists of first clustering companies with similar behaviour and afterwards, employing the NN approach, training and testing each clustered data separately. The NN learning unveiled the patterns behind each cluster of companies obtained using unsupervised clustering. Results indicated which companies (or clusters “outliers”) would be worth investigating thoroughly. We are now following an in site inspection plan on those companies to verify whether they show some type of irregularity in their accountancy. Other companies within each cluster will be inspected using statistical sampling.

The methodology then consisted of clustering the database to find groups of companies with similar behaviour; then use Neural Network training and testing to find the function that best explain the behaviour of the cluster. Using a system based on this approach requires first classifying the company in one of the clusters, and then using the trained Neural Network to forecast the tax due.

Further to evaluating the results by means of in site inspections, we will also employ traditional Regression techniques on clustered data so as to compare results with those from Artificial Neural Networks.

7 ACKNOWLEDGEMENTS

The authors would like to thank Brazilian Lutheran University and FAPERGS for providing financial support for this research. We are also grateful to the Brazilian Government Body that provided the data for analysis.

8 REFERENCES

- [1]. De Carvalho, M. C. A Comparison of Neural Networks and Econometric Discrete Choice Models in Transport. *Unpublished PhD Thesis*, ITS, university of Leeds, UK, 1998.
- [2]. Rumelhart, D. E., Hinton, G. E. and Williams, R. J. Learning internal representations by error propagation. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Vol. 1: Foundations, D.E. Rumelhart, J.L.McLelland, and PDP group, Eds. MIT Press, Cambridge, MA, 1986.
- [3]. Hecht-Nielsen, R. (1989). Theory of the backpropagation neural network. *Proceedings of the International Joint Conference on Neural Networks*, IEEE Press, New York.
- [4]. Bishop, C. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
- [5]. Jain, A.K.; M.N.Murty; P.J.Flynn. Data Clustering: A Review. *ACM Computing Surveys*, Vol. 31, N.3, 1999.
- [6]. Bock, H. H. Probabilistic models in cluster analysis. *Computational Statistics & Data Analysis* 23,1996.
- [7]. Michaud, P. Clustering Techniques. *Future Generation Computer Systems* 13, 1997.
- [8]. Cheesmam, P., J. Stutz. Bayesian Classification (Autoclass): Theory and Results. *NASA*, 1995. Available with Autoclass package at (<http://ic-www.arc.nasa.gov/ic/projects/bayes-group/autoclass/>).

Self-Organized Data and Image Retrieval as a Consequence of Inter-Dynamic Synergistic Relationships in Artificial Ant Colonies

Vitorino RAMOS, Fernando MUGE, Pedro PINA
 CVRM – GeoSystems Centre, Technical Univ. of Lisbon (IST),
 Av. Rovisco Pais, Lisbon, 1049-001, Portugal

Abstract. Social insects provide us with a powerful metaphor to create decentralized systems of simple interacting, and often mobile, agents. The emergent collective intelligence of social insects – swarm intelligence – resides not in complex individual abilities but rather in networks of interactions that exist among individuals and between individuals and their environment. The study of ant colonies behavior and of their self-organizing capabilities is of interest to knowledge retrieval/ management and decision support systems sciences, because it provides models of distributed adaptive organization which are useful to solve difficult optimization, classification, and distributed control problems, among others. In the present work we overview some models derived from the observation of real ants, emphasizing the role played by stigmergy as distributed communication paradigm, and we present a novel strategy (ACLUSTER) to tackle unsupervised data exploratory analysis as well as data retrieval problems. Moreover and according to our knowledge, this is also the first application of ant systems into digital image retrieval problems. Nevertheless, the present algorithm could be applied to any type of numeric data.

1. Introduction: Stigmergy and Distributed Awareness

Synergy, from the greek word *synergos*, broadly defined, refers to combined or co-operative effects produced by two or more elements (parts or individuals). The definition is often associated with the quote “the whole is greater than the sum of its parts” (Aristotle, in *Metaphysics*), even if it is more accurate to say that the functional effects produced by wholes are different from what the parts can produce alone. Synergy is a ubiquitous phenomena in nature and human societies alike. One well know example is provided by the emergence of self-organization in social insects, via direct (mandibular, antennation, chemical or visual contact, etc) or indirect interactions. The latter types are more subtle and defined by *Grassé* as stigmergy [5] to explain task coordination and regulation in the context of nest reconstruction in *Macrotermes* termites. An example [1], could be provided by two individuals, who interact indirectly when one of them modifies the environment and the other responds to the new environment at a later time. In other words, stigmergy could be defined as a typical case of environmental synergy. *Grassé* showed that the coordination and regulation of building activities do not depend on the workers themselves but are mainly achieved by the nest structure: a stimulating configuration triggers the response of a termite worker, transforming the configuration into another configuration that may trigger in turn another (possibly different) action performed by the same termite or any other worker in the colony. Another illustration of how stigmergy and self-organization can be combined into more subtle adaptive behaviors is recruitment in social insects. Self-organized trail laying by individual ants is a way of modifying the environment to communicate with nest mates that follow such trails [1]. Division of labor is also another paradigmatic phenomena of stigmergy. Simultaneous task performance (parallelism) by specialized workers is believed to be more efficient than sequential task performance by unspecialized workers. But by far more crucial to the present work and aim, is how ants form piles of items such as dead bodies (corpses), larvae, or grains of sand (fig. 1). There again, stigmergy is at work: ants deposit items at initially random locations. When other ants perceive deposited items, they are stimulated to deposit items next to them, being this type of cemetery clustering action,

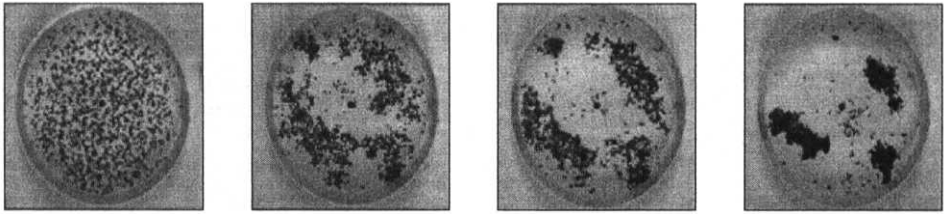


Figure 1. From left to right, a sequential clustering task of corpses performed by a real ant colony. 1500 corpses are randomly located in a circular arena with radius = 25 cm, where *Messor Sancta* workers are present. The fig. shows the initial state (left), 2 hours, 6 hours and 26 hours (right) after the beginning of the experiment [1].

organization, and brood sorting a type of self-organization and adaptive behavior. There are other types of examples (e.g. prey collectively transport), yet stigmergy is also present: ants change the perceived environment of other ants (their cognitive map, according to [3]), and in every example, the environment serves as medium of communication [1]. Nevertheless, what all these examples have in common is that they show how stigmergy can easily be made operational. As mentioned by *Bonabeau* in [1], that is a promising first step to design groups of artificial agents which solve problems: replacing coordination (and possibly some hierarchy) through direct communications by indirect interactions is appealing if one wishes to design simple agents and reduce communication among agents. Finally, stigmergy is often associated with flexibility: when the environment changes because of an external perturbation, the insects respond *appropriately* to that perturbation, as if it were a modification of the environment caused by the colony's activities. When it comes to artificial agents, this type of flexibility is priceless: it means that the agents can respond to a perturbation without being reprogrammed to deal with that particular instability. In our context, this means that no classifier re-training is needed for any new sets of data-item types (new classes) arriving to the system, as is necessary in many classical models, or even in some recent ones. Moreover, the data-items that were used for supervised purposes in early stages, can now, along with new items, be re-arranged in more optimal ways. Classification and/or data retrieval remains the same, but the system reorganizes itself in order to deal with new classes, or even new sub-classes. Recently, several papers (for a good revision see [1]) have highlighted the efficiency of stochastic approaches based on ant colonies for different problem solving. Data clustering is also one of those problems in which real ants can suggest very interesting heuristics for computer scientists. One of the first studies using the metaphor of ant colonies related to the above clustering domain is due to *Deneubourg* [4], where a population of ant-like agents randomly moving onto a 2D grid are allowed to move basic objects so as to cluster them. This method was then further generalized by *Lumer et al.* [10], applying it to exploratory data analysis, for the first time. Our aim is to improve these models, introducing some radical changes and different ant-like heuristics, developing a model without any local memory and/or hybridization with more classical approaches. Moreover, the present work will be applied for the first time to image retrieval and exploratory data analysis. The datasets represent a collection of the most representative 14 types of Portuguese grey granites (fig.3), with a total set of 237 images, each represented by 117 Mathematical Morphology [16,9,12] features. Some chinese granites were also used as a test, since they can mislead several human experts, leading to a total of 244 images \times 177 features. Sections III -IV describe the present proposal, while results are in sections V-VI.

2. Corpse Clustering and Variants into Exploratory Data Analysis

In several species of ants, workers have been reported to sort their larvae or form piles of corpses – literally cemeteries – to clean up their nests. *Chrétien* (see [1]) has performed experiments with the ant *Lasius niger* to study the organization of cemeteries. Other

experiments include the ants *Pheidole pallidula* reported in [4] by Denebourg et al., and many species actually organize a cemetery. Figure 1 (section I) shows the dynamics of cemetery organization in another species: *Messor sancta*. If corpses, or more precisely, sufficiently large parts of corpses are randomly distributed in space at the beginning of the experiment, the workers form cemetery clusters within a few hours, following a behavior similar to aggregation. If the experimental arena is not sufficiently large, or if it contains spatial heterogeneities, the clusters will be formed along the edges of the arena or, more generally, following the heterogeneities. The basic mechanism underlying this type of aggregation phenomenon is an attraction between dead items mediated by the ant workers: small clusters of items grow by attracting workers to deposit more items. It is this positive and auto-catalytic feedback that leads to the formation of larger and larger clusters. In this case, it is therefore the distribution of the clusters in the environment that plays the role of stigmergic variable. Denebourg et al. [4] have proposed one model (hereafter called BM, for basic model) to account for the above-mentioned phenomenon of corpse clustering in ants. The general idea is that isolated items should be picked up and dropped at some other location where more items of that type are present. Let us assume that there is only one type of item in the environment. The probability P_p for a randomly moving, unladen agent (representing an ant in the model) to pick up an item is given by (Eq. 2.1):

$$P_p = \left(\frac{k_1}{k_1 + f} \right)^2 \quad (2.1)$$

$$P_d = \left(\frac{f}{k_2 + f} \right)^2 \quad (2.2)$$

where f is the perceived fraction of items in the neighborhood of the agent, and k_1 is a threshold constant. When $f \ll k_1$, P_p is close to 1, that is, the probability of picking up an item is high when there are not many items in the neighborhood. P_p is close to 0 when $f \gg k_1$, that is, items are unlikely to be removed from dense clusters. The probability P_d for a randomly moving loaded agent to deposit an item is given by Eq. 2.2., where k_2 is another threshold constant: for $f \ll k_2$, P_d is close to 0, whereas for $f \gg k_2$, P_d is close to 1. In their simulations, Denebourg et al. [4] have used $k_1 = 0.1$ and $k_2 = 0.3$, testing the spatial sorting organization of 400 items of two types, on a 100×100 grid, using 10 agents and $T = 50$; 5,000,000 iterations were needed to accomplish a feasible visual result. As expected, the depositing behavior obeys roughly opposite rules.

As we shall see, the algorithms later described (as well as those proposed) in the present work, are inspired by this idea, but rely on a more direct evaluation of f . This procedure should, therefore, be taken as an example among many possible procedures, and changing the detail how f is perceived does not drastically alter the results, according to Bonabeau [1]. Among other differences proposed later, are also those directly related to how the agents move on the spatial grid. For instance, real ants are likely to use chemical or tactile cues to orient their behavior. In their simulations, however, Denebourg et al. [4] have taken the option of using randomly moving agents, while in here and due to our aim, we suggest the use of ant-like spatial transition probabilities (section III), based on chemical pheromone non-linear weighting functions. Significantly more interesting to the present proposal is however, Lumer's and Faieta model [10]. Both authors have generalized Denebourg et al.'s BM [4], to apply it to exploratory data analysis. The idea is to define a distance or dissimilarity d between objects in the space of object attributes. Let $d(o_i, o_j)$ be the distance between two objects o_i and o_j in the feature space. Let us also assume that an agent is located at site r at time t , and finds an object o_i at that site. The "local density" $f(o_i)$ with respect to object o_i at site r is then given by:

$$f(o_i) = \max \left\{ 0, \frac{1}{s^2} \sum_{o_j \in \text{Neigh}_{(1\text{er})}(r)} \left[1 - \frac{d(o_i, o_j)}{\alpha} \right] \right\} \quad (2.3)$$

$f(o_i)$ is a measure of the average similarity of object o_i with the other objects o_j present in the neighborhood of o_i . That is, $f(o_i)$ replaces the fraction f of similar objects in the BM model, while α is a factor that defines the scale of dissimilarity: it is important for it determines when two items should or should not be located next to each other. Then, and inspired by Denebourg et al.'s functions [4] (Eqs. 2.1 and 2.2), Lumer and Faieta [10] defined picking up probabilities similarly, where f was substituted by $f(o_i)$, and dropping probabilities as $P_d = 2 \cdot f(o_i)$ for $f(o_i) < k_2$, and $P_d = 1$ for the remaining cases. Lumer and Faieta [10] have used $k_1 = 0.1$, $k_2 = 0.15$ (while BM uses $k_2 = 0.3$) and $\alpha = 0.5$, with $t_{\max} = 10^6$ steps. In order to illustrate the functioning of their algorithm, the authors used a simple example in which the attribute space is R^2 , and the values of the two attributes for each object correspond to its coordinates (x, y) in R^2 . The same distribution was later used for tests in the present work – see fig. 2a, section V. Lumer and Faieta [10] have then added three features to their system, due to the fact that there are generally more clusters in the projected system than in the initial distribution. These features help to solve this problem, even if they are computationally intensive and broadly bio-inspired. They are: (1) ants with different moving speeds, (2) a short term memory, and (3), behavioral switches.

3. From Randomly Moving Agents to Bio-Inspired Spatial Probabilities

Instead of trying to solve some disparities in the basic LF algorithm by adding different ant casts, short-term memories and behavioral switches (described in section II) which are computationally intensive, representing simultaneously a potential and difficult complex parameter tuning, it is our intention (within the present ACLUSER proposal) to follow real ant-like behaviors as possible (some other features will be incorporated, as the use of different response thresholds to task-associated stimulus intensities, discussed later at section IV). In that sense, bio-inspired spatial transition probabilities are incorporated into the system, avoiding randomly moving agents, which tend the distributed algorithm to explore regions manifestly without interest (e.g., regions without any type of object clusters), being generally, this type of exploration, counterproductive and time consuming. Since this type of transition probabilities depend on the spatial distribution of pheromone across the environment, the behavior reproduced is also a stigmergic one. Moreover, the strategy not only allows to guide ants to find clusters of objects in an adaptive way (if, by any reason, one cluster disappears, pheromone tends to evaporate on that location), as the use of embodied short-term memories is avoided (since this transition probabilities tends also to increase pheromone in specific locations, where more objects are present). As we shall see, the distribution of the pheromone represents the memory of the recent history of the swarm, and in a sense it contains information which the individual ants are unable to hold or transmit. There is no direct communication between the organisms but a type of indirect communication through the pheromonal field. In fact, ants are not allowed to have any memory and the individual's spatial knowledge is restricted to local information about the whole colony pheromone density. In order to design this behavior, one simple model was adopted (Chialvo and Millonas, [3]), and extended (as in [13]) due to specific constraints of the present proposal. As described in [3], the state of an individual ant can be expressed by its position r , and orientation θ . It is then sufficient to specify a transition probability from one place and orientation (r, θ) to the next (r^*, θ^*) an instant later. The response function can effectively be translated into a two-parameter transition rule between the cells by use of a pheromone weighing function (Eq. 3.1):

$$W(\sigma) = \left(1 + \frac{\sigma}{1 + \delta\sigma}\right)^\beta \quad (3.1) \quad P_{ik} = \frac{W(\sigma_i)w(\Delta_i)}{\sum_{j/k} W(\sigma_j)w(\Delta_j)} \quad (3.2)$$

This equation measures the relative probabilities of moving to a cite r (in our context, to a grid location) with pheromone density $\sigma(r)$. The parameter β is associated with the osmotropotaxic sensitivity (a kind of instantaneous pheromonal gradient following), and on the other hand, $1/\delta$ is the sensory capacity, which describes the fact that each ant's ability to sense pheromone decreases somewhat at high concentrations. In addition to the former equation, there is a weighing factor $w(\Delta\theta)$, where $\Delta\theta$ is the change in direction at each time step, i.e. measures the magnitude of the difference in orientation. As an additional condition, each individual leaves a constant amount η of pheromone at the pixel in which it is located at every time step t . This pheromone decays at each time step at a rate k . Then, the normalised transition probabilities on the lattice to go from cell k to cell i are given by P_{ik} [3] (Eq. 3.2), where the notation j/k indicates the sum over all the pixels j which are in the local neighbourhood of k . Δ_i measures the magnitude of the difference in orientation for the previous direction at time $t-1$. That is, since we use a neighbourhood composed of the cell and its eight neighbours, Δ_i can take the discrete values 0 through 4, and it is sufficient to assign a value w_i for each of these changes of direction. *Chialvo et al* used the weights of $w_0=1$ (same direction), $w_1=1/2$, $w_2=1/4$, $w_3=1/12$ and $w_4=1/20$ (U-turn). In addition, coherent results were found for $\eta=0.07$ (pheromone deposition rate), $k=0.015$ (pheromone evaporation rate), $\beta=3.5$ (osmotropotaxic sensitivity) and $\delta=0.2$ (inverse of sensory capacity), where the emergence of well defined networks of trails were possible. For a detailed mathematical discussion of this model, and other conditions readers are reported to [3] and [13]. In order to achieve emergent and *autocatalytic* mass behaviours around item groups on the *habitat*, which can significantly change the expected ant colony cognitive map (pheromonal field), instead of a constant pheromone deposition rate η used in [3], a term not constant is included. This strategy follows an idea implemented by *Ramos et al.* [13], while extending the *Chialvo* model into digital image habitats. In here, however, this term is naturally related with the amount of items found in one specific region. So for instance, if we use Δ_h as that measure (i.e., the number of items present in one neighborhood), the pheromone deposition rate T for a specific ant at that specific cell (at time t), should change to a dynamic value (p is a constant = 0.0025): $T = \eta + p\Delta_h$. Notice that, if no objects are present, results expected by this extended model will be equal to those found by *Chialvo* and *Millonas* in [3], since Δ_h equals to zero.

4. Stressing the Role of Response Thresholds to Task-Associated Stimulus Intensities

In order to model the behavior of ants associated to different tasks, as dropping and picking up objects, we suggest the use of combinations of different response thresholds. As we have seen before, there are two major factors that should influence any local action taken by the ant-like agent: the number of objects in his neighborhood, and their similarity (including the hypothetical object carried by one ant). *Lumer and Faieta* [10], use an average similarity (Eq. 2.3, section II), mixing distances between objects with their number, incorporating it simultaneously into a response threshold function like the one of *Denebourg's* (Eq. 2.1, 2.2, section II). Instead, in the present proposal, we suggest the use of combinations of two independent response threshold functions, each associated with a different environmental factor (or, stimuli intensity), that is, the number of objects in the area, and their similarity. Moreover, the computation of average similarities are avoided in the present algorithm, since

this strategy can be somehow blind to the number of objects present in one specific neighborhood. In fact, in *Lumer* and *Faieta*'s work [10], there is an hypothetical chance of having the same average similarity value, respectively having one or, more objects present in that region. But, experimental evidences and observation in some types of ant colonies, can provide us with a different answer. After *Wilson* [17], it is knowned that minors and majors in the polymorphic species of ants *Genus Pheidole*, have different response thresholds to task-associated stimulus intensities (i.e., division of labor). Recently, and inspired by this experimental evidence, *Bonabeau* et al. [2], proposed a family of response threshold functions in order to model this behavior. According to it, every individual has a response threshold θ for every task. Individuals engage in task performance when the level of the task-associated stimuli s , exceeds their thresholds. Authors defined s as the intensity of a stimulus associated with a particular task, i.e. s can be a number of encounters, a chemical concentration, or any quantitative cue sensed by individuals. One family of response functions $T_\theta(s)$ (the probability of performing the task as a function of stimulus intensity s), that satisfy this requirement is given by (Eq. 4.1) [2]:

$$T_\theta(s) = \frac{s^n}{s^n + \theta^n} \quad (4.1)$$

$$\chi = \frac{n^2}{n^2 + s^2} \quad (4.2)$$

$$\delta = \left(\frac{k_1}{k_1 + d} \right)^2 \quad (4.3)$$

$$\varepsilon = \left(\frac{d}{k_2 + d} \right)^2 \quad (4.4)$$

where $n > 1$ determines the steepness of the threshold (normally $n=2$, but similar results can be obtained with other values of $n > 1$). Now, at $s = \theta$, this probability is exactly $1/2$. Therefore, individuals with a lower value of θ are likely to respond to a lower level of stimulus. In order to take account on the number of objects present in one neighborhood, Eq. 4.1, was used (where, n now stands for the number of objects present in one neighborhood, and $\theta = 5$), defining χ (Eq. 4.2) as the response threshold associated to the number of items present in a 3×3 region around r (one specific grid location). Now, in order to take account on the hypothetical similarity between objects, and in each ant action due to this factor, a Euclidean normalized distance d is computed within all the pairs of objects present in that 3×3 region around r . Being a and b , a pair of objects, and $f_a(i), f_b(i)$ their respective feature vectors (being each object defined by F features), then $d = (1/d_{max}) \cdot [(1/F) \cdot \sum_{i=1, F} (f_a(i) - f_b(i))^2]^{1/2}$. Clearly, this distance d reaches its maximum ($=1$, since d is normalized by d_{max}) when two objects are maximally different, and $d=0$ when they are equally defined by the same F features. Then, δ and ε (Eqs. 4.3, 4.4), are respectively defined as the response threshold functions associated to the similarity of objects, in case of dropping an object (Eq. 4.3), and picking it up (Eq. 4.4), at site r . Note that these functions are similar to those proposed by *Denebourg* et al. [4] (k_1 and k_2 , are threshold constants), while defining probabilities for picking up or to deposit an item (Eqs. 2.1, 2.2, section II). In here, however, we use them in reversed order, substituting f by d (where f represented, for *Denebourg* et al., the perceived fraction of items in the neighborhood of one agent, having in mind a robotic implementation). As we can observe, the probability δ for a specific moving loaded agent to deposit an item at site r , is given by Eq. 4.3. When $d \ll k_1$ (i.e., d close to 0), δ is close to 1, that is, the probability of dropping an item is high when the similarity between the loaded object and one present in the region around r , is high.

Now, in every action taken by an agent, and in order to deal, and represent different stimulus intensities (number of items and their similarity), present at each site in the environment visited by one ant, the strategy uses a composition of the above defined response threshold functions (Eq. 4.2, 4.3 and 4.4). These composed probabilities are resumed in table 1, and were used as test functions in one preliminar test (section V) proposed in [10] in order to illustrate the functioning of the algorithm.

TYPES OF HYBRID RESPONSE FUNCTIONS USED				
Function Types	Picking Probability		Dropping Probability	
#1	$P_p = (1-\chi) \cdot \epsilon$		$P_d = \chi \cdot \delta$	
#2	(a) $P_p = (1-\chi) \cdot \epsilon$	(b) $P_p = \epsilon$	(a) $P_d = \chi \cdot \delta$	(b) $P_d = \delta$
#3	(a) $P_p = 1-\chi$	(b) $P_p = \epsilon$	(a) $P_d = \chi$	(b) $P_d = \delta$
#4	Lumer & Faieta (see section II)		Lumer & Faieta (see section II)	

Table 1 – Types of picking (P_p) and dropping (P_d) probability functions used for several tests. In #2,3 half of the ants used one probability function (a), while the rest used the other function (b). In #4, the LF algorithm (section II) was fully implemented and followed, but using a toroidal grid.

On the other hand, to evaluate the algorithm behavior, a simple entropy definition is proposed. For a finite number of n type A items, placed into a finite area grid, the entropy of A type objects can be defined as the normalized sum, over all n , of the number of empty cells e (or occupied by objects different from A), surrounding each item A ($e_{max} = 8$, in 3×3 regions), that is, $E_A = (\sum e_i) / (n \cdot e_{max})$. As its obvious, several configurations lead to different values of entropy, where E_A reaches its maximal value ($E_A = 1$) when all type A items are disconnected from each other. Disconnected clusters of type A items, lead also to an increase in the value of entropy.

5. Results on a “4 Classes X 200 Gaussian Distributed Points” Problem

As mentioned before, we decide to test the algorithm using the same problem as *Lumer and Faieta*, introduced by them in [10]. This problem consists of 800 points, represented by two features each. That is, the attribute space is R^2 , and the values of the two attributes for each object correspond to its coordinates (x,y) in R^2 . Four clusters of 200 points each were then generated in attribute space, with x and y distributed according to Normal (or Gaussian) distributions $N(\mu,\sigma)$ of average μ and variance σ^2 - see figure 2.a) for details. The 800 data points (items) were then assigned at random locations on a 57×57 non-parametric toroidal grid, and the clustering algorithm was run with 80 ants, using the function types specified in table 1. Generally, the following empirical rules were followed, since they lead to good results: $A=4 \cdot n_o$, $n_a=A/40$, and $n_a/n_o=0.1$, where A is the grid area, n_o is the number of objects, and n_a the number of ants used. As a final result, objects that are clustered together belong generally to the same initial distribution, and objects that do not belong to the same initial distribution are found generally in different clusters. In figure 2.b), the evolution of total entropy ($E_{total}=E_A+E_B+E_C+E_D$), for 10^6 iterations (as those used in [10]) was plotted for four different type functions. It is clear to see that probabilistic functions type #3, are the worse in terms of clustering the different items, while the rest (including the algorithm proposed by *Lumer and Faieta* [10]) have similar behaviors, and indeed reduce drastically the value of entropy of those configurations. We can also get an idea of how the new algorithm clusters the different items, while the algorithm proceeds (fig. 2.c,d,e,f). In this case type function #1 was used, and as observed, initially randomly deposited items at $t=1$ in the toroidal grid, are then at $t>0$ spatially distributed according to their similarities.

6. Applications into Digital Image Retrieval: A Case Study within a Granite Database

Ornamental stones are quantitatively characterised in many ways, mostly physical, namely, geological-petrographical and mineralogical composition, or by mechanical strength. However, the properties of such products differ not only in terms of type but also in terms of origin, and their variability can also be significant within a same deposit or quarry [12].

Algorithm. High-level description of *ACLUSTER*.

```

/* Initialization */
For every item  $o_i$  do
Place  $o_i$  randomly on grid
End For
For all agents do
Place agent at randomly selected site
End For
/* Main loop */
For  $t = 1$  to  $t_{max}$  do
For all agents do
 $sum = 0$ 
Count the number of items  $n$  around site  $r$ 
If ((agent unladen) and (site  $r$  occupied by item  $o_i$ )) then
For all sites around  $r$  with items present do
/* According to Eqs. 4.2, 4.4 and Table 1 (section 4) */
Compute  $d, \chi, \varepsilon$  and  $P_p$ 
Draw random real number  $R$  between 0 and 1
If ( $R \leq P_p$ ) then
 $sum = sum + 1$ 
End If
End For
If ( $sum \geq n/2$ ) or ( $n = 0$ ) then
Pick up item  $o_i$ 
End If
Else If ((agent carrying item  $o_i$ ) and (site  $r$  empty)) then
For all sites around  $r$  with items present do
/* According to Eqs. 4.2, 4.3 and Table 1 (section 4) */
Compute  $d, \chi, \delta$  and  $P_d$ 
Draw random real number  $R$  between 0 and 1
If ( $R \leq P_d$ ) then
 $sum = sum + 1$ 
End If
End For
If ( $sum \geq n/2$ ) then
Drop item  $o_i$ 
End If
End If
/* According to Eqs. 3.1 and 3.2 (section 3) */
Compute  $W(r)$  and  $P_{is}$ 
Move to a selected  $r$  not occupied by other agent
Count the number of items  $n$  around that new site  $r$ 
Increase pheromone at site  $r$  according to  $n$ , that is:
 $P_r = P_r + [\eta + (n/\alpha)]$ 
End For
Evaporate pheromone by  $K$ , at all grid sites
End For
Print location of items
/* Values of parameters used in experiments */
 $k_1 = 0.1, k_2 = 0.3, K = 0.015, \eta = 0.07, \alpha = 400,$ 
 $\beta = 3.5, \gamma = 0.2, t_{max} = 10^6$  steps.

```

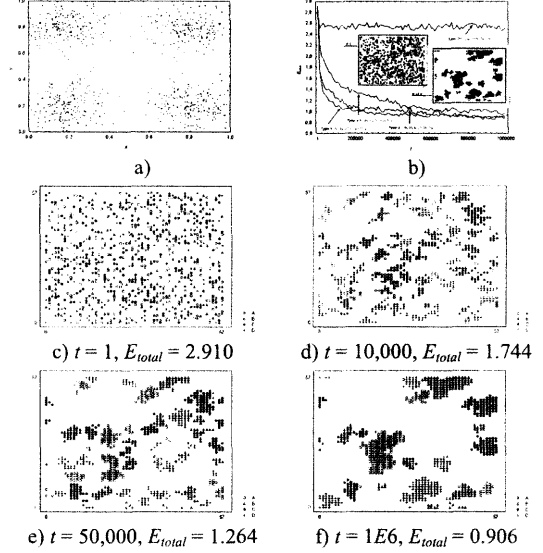


Fig. 2. **a)** Distribution of points in “attribute space”: 4 clusters of 200 points each were generated in attribute space, with x and y distributed according to Normal (or Gaussian) distributions $N(\mu, \sigma)$: $A = [x \approx N(0.2, 0.1), y \approx N(0.2, 0.1)]$, $B = [x \approx N(0.8, 0.1), y \approx N(0.2, 0.1)]$, $C = [x \approx N(0.8, 0.1), y \approx N(0.8, 0.1)]$, $D = [x \approx N(0.2, 0.1), y \approx N(0.8, 0.1)]$, for objects type A, B, C and D , respectively. **b)** Total entropy, $E_{total} = E_a + E_b + E_c + E_d$, in time, as the swarm evolves new solutions in clustering four type of objects. Four graphs are shown which correspond to four types of Probability functions (dropping and picking) analyzed (see table 1). **c, d, e, f)** Spatial distribution of 800 items on a 57×57 non-parametric toroidal grid at several time steps. At $t=1$, four types of items are randomly allocated into the grid. As time evolves, several homogenous clusters emerge due to the ant colony action, and as expected the total entropy decreases. In order to illustrate the behavior of the algorithm, items that belong to different clusters (fig. 2.a), were in here represented by different symbols: \circ, Δ, \bullet and $+$. Type 1 probability function was used with $k_1=0.1$ and $k_2=0.3$.

Though useful, these methods do not fully solve the problem of classifying a product whose end-use makes appearance so critically important. Appearance is conditioned not only by the kind of stone but also depends on the subjective evaluation of beauty and hence of economic value, which are strongly influenced by supply and demand. Traditionally, the selection process is based on visual inspection, giving a subjective characterisation of the appearance of the materials. Thus, one suitable tool to characterise the appearance of these natural stones is digital image analysis. In the present work we use mathematical morphology (MM, [16]) as a feature extraction method, as used in past works by Ramos et al. [12]. Generically, the extraction of features by means of image analysis and mathematical morphology techniques is implemented in 2 stages: a global and a local analysis. It consists on the extraction of features before (global analysis) and after (local analysis) the segmentation or mineral phase classification procedures. This approach is general and can be applied not only to characterise slab surfaces of granites as also other types of ornamental stones. The method fully described

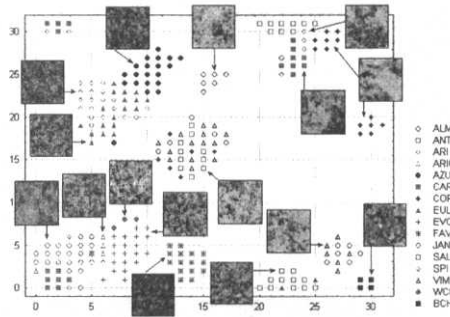


Figure 3 – Spatial distribution of 244 images (representing 14 types of Portuguese Granites + 2 types of Chinese Granites), at $t=1,000,000$. Each image (point in the environment) is composed by 117 morphological and intensity features. Type 1 probability function was used with $k_1=0.1$ and $k_2=0.3$.

in [12] uses the opening MM operator (erosion followed by dilation) and the closing (dilation followed by erosion), since they have granulometric properties [11] once are increasing, extensive (closing) and anti-extensive (opening) and idempotent. From here, a size-intensity diagram [9], can be extracted and used as a feature vector for each image. Data was collected for 14 types of Portuguese grey granites, with a total set of 237 images, each represented by 117 MM features. Some chinese granites were also used as a test, since they can mislead several human experts, leading to a total of 244 images \times 177 features. Since we had 244 items (images) to self-organize by the swarm, 24 ants were used (see section V), on a 31×31 non-parametric toroidal grid. Fig. 3 shows the final result at $t=10^6$, as well as the type of granite textures clustered.

7. Conclusions

We have presented in this paper a new ant-based algorithm named ACLUSTER for data unsupervised clustering and data exploratory analysis, while sketching a clear parallel between a mode of problem-solving in social insects and a distributed, reactive, algorithmic approach. Some of the mechanisms underlying corpse clustering, brood sorting and those that can explain the worker's behavioral flexibility, as regulation of labor and allocation of tasks have first been introduced. As in similar past works applied to document clustering and text retrieval [14], the role of response thresholds to task-associated stimulus intensities were stressed as an important part of the strategy, and incorporated into the algorithm by using compositions of different response functions. These compositions allows the strategy not only to be more accurate relatively to behaviours found in nature as avoids short-term memory based strategies, and the use of several artificial ant types (using different speeds), present in some recent approaches. Behavioral switches as used in [10], were also avoided, in order to maintain simplicity and to avoid complex parameter settings to be performed by the domain expert. At the level of agent moves in the grid, a truly stigmergic model was introduced (section III) in order to deal with clusters of objects, avoiding randomly moves which can be counterproductive in the distributed search performed by the swarm, and adopted by all past models. In fact, the present algorithm, along with [14], were the first to introduce pheromone traces on agents to deter random explorations and encourage objects cluster formation, a successful feature not implemented even in some recent proposals [7,6]. Results speak for themselves (fig.2,3). While achieving similar results compared to *Lumer's* model [10], as pointed by the spatial entropy of solutions at each time iteration (fig.2b), the present algorithm is by far more simple. Moreover, for some of response thresholds compositions used, results are superior while using the present algorithm for the majority of time iterations, that is, entropy is always lower, even if at the end they tend to the same

value. As a final advantage, ACLUSTER does not require any initial information about the future classification, such as an initial partition or an initial number of classes. This novel strategy was then applied for the first time to digital image retrieval via k -NNR. Generally, similar types of images tend to be homogeneously clustered together. But more impressive is that this type of stigmergic map can be used to classify new images, arriving to the system at any moment. In fact, using 50 randomly chosen images as a test set (from the initial 244; several sub-sets were used), an average classification and retrieval rate of 94% was achieved by using k -NNR classification methods (*Nearest Neighbor Rule* – i.e., a label of an unknown item is determined by the label of his first k neighbors; $k=3$ was used). Finally, and as verified by other tests [15] on ACLUSTER, a robust nonstop classifier could be achieved, which produces class decisions on a continuous stream data, allowing for continuous mappings. As we know, many categorization systems have the inability to perform classification and visualization in a continuous basis or to self-organize new data-items into the older ones (evenmore into new labels if necessary), unless a new training happens. This disadvantage is also present in more recent approaches using Self-Organizing Maps [8], as in *Kohonen* maps. While a benchmark comparison of the above cited methods should be interesting to explore, the ability of ACLUSTER to perform continuous mappings and the incapacity of the latter to conceive it, tend to difficult any serious comparison.

References & Acknowledgements: The first author wishes to thank to FCT-PRAXIS XXI (BD20001-99) - *Ministério da Ciência e do Ensino Superior*, for his Research Fellowship.

- [1] Bonabeau E., M. Dorigo, G. Théraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Santa Fe Institute in the Sciences of the Complexity, Oxford University Press, New York, Oxford, 1999.
- [2] Bonabeau, E., Théraulaz, G., Deneubourg, J.-L., "Quantitative Study of the Fixed Response Threshold Model for the Regulation of Division of Labour in Insect Societies". *Roy. Soc. B*, 263, pp. 1565-1569, 1996.
- [3] Chialvo, Dante R., Millonas, Mark M. "How Swarms Build Cognitive Maps". In Luc Steels (Ed.), *The Biology and Technology of Intelligent Autonomous Agents*, (144) pp. 439-450, NATO ASI Series, 1995.
- [4] Deneubourg, J.-L., Goss, S., Franks, N., Sendova-Franks A., Detrain, C., Chretien, L. "The Dynamic of Collective Sorting Robot-like Ants and Ant-like Robots", *SAB'90 - 1st Conf. On Simulation of Adaptive Behavior: From Animals to Animats*, J.A. Meyer and S.W. Wilson (Eds.), 356-365. MIT Press, 1991.
- [5] Grassé, P.-P. "Termitologia, Tome II" *Fondation des Sociétés. Construction*. Paris; Masson, 1984.
- [6] Handl, J., Meyer B. "Improved Ant-based Clustering and Sorting in a Document Retrieval Interface", in J.J. Merelo, J.L.F. Villacañas, H.-G. Beyer, P. Adamidis (Eds.), *PPSN VII - 7th Int. Conf. on Parallel Problem Solving from Nature*, Granada, Spain, Sept. 2002.
- [7] Hoe, K.M., Lai, W.K., Tai, T.S.Y. "Homogeneous Ants for Web Document Similarity Modeling and Categorization", in M. Dorigo, G. Di Caro and M. Sampels (Eds.), *3rd Int. Works. on Ant Algorithms*, Springer-Verlag, Brussels, Sept. 2002.
- [8] Kohonen, T. *Self-Organizing Maps*, Springer-Verlag, Berlin, 1997.
- [9] Lotufo, R.A., Trettel, E., "Integrating Size Information into Intensity Histogram", in Maragos P., Schafer R.W., Butt M.A.(Eds.), *Mathematical Morphology and its Applications to Image and Signal Processing*, pp. 281-288, Kluwer Academic Publishers, Boston, 1996.
- [10] Lumer E. D. & Faieta B. (1994), Diversity and Adaptation in Populations of Clustering Ants. In Cliff, D., Husbands, P., Meyer, J. and Wilson S. (Eds.), in *From Animals to Animats 3*, Proc. of the 3rd Int. Conf. on the Simulation of Adaptive Behavior. Cambridge, MA: The MIT Press/Bradford Books, 1994.
- [11] Matheron G. *Random sets and Integral Geometry*, Wiley, New York, 1975.
- [12] Ramos, Vitorino, Pina, P. and Muge, F. "From Feature Extraction to Classification: A Multidisciplinary Approach applied to Portuguese Granites", *11th Scan. Conf. on Image Analysis*, pp. 817-824, Greenland, 1999.
- [13] Ramos, Vitorino, and Almeida, F. "Artificial Ant Colonies in Digital Image Habitats - A Mass Behaviour Effect Study on Pattern Recognition", in Marco Dorigo, Martin Middendorf and Thomas Stützle (Eds.), *Proc. of ANTS'00 - 2nd Int. Workshop on Ant Algorithms*, pp. 113-116, Brussels, Belgium, 7-9 Sep. 2000.
- [14] Ramos, Vitorino and Merelo, Juan J. "Self-Organized Stigmergic Document Maps: Environment as a Mechanism for Context Learning", in E. Alba, F. Herrera, J.J. Merelo et al. (Eds.), *AEB'02 - 1st Int. Conf. on Metaheuristics, Evolutionary and Bio-Inspired Algorithms*, pp. 284-293, Mérida, Spain, 6-8 Feb. 2002.
- [15] Ramos, Vitorino, "Swarms on Continuous Data", to appear in Ahmad Lotfi, Bob John and Jon Garibaldi (Eds.), *Recent Advances in Soft Computing*, Springer-Verlag, Nottingham, UK, 12-13 Dec. 2002.
- [16] Serra J., *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.
- [17] Wilson, E.O., *The Insect Societies*, Belknap Press, Cambridge, 1971.

This page intentionally left blank

Section 9

Agent Architectures

This page intentionally left blank

Designing Not-So-Dull Virtual Dolls

Fabio Zambetta, Graziano Catucci, SAMIR

V.A.L.I.S. Group

Dipartimento di Informatica, Università degli Studi, Via E. Orabona 4, Bari, I-70124

Abstract. Intelligent virtual agents exhibiting autonomous behavior rather than mere reactions to users actions are going to become a major requirement for modern web sites. In this paper we present SAMIR, a system conceived to design intelligent agents with a 3D animated look as a front-end, to enhance the user interaction with the web applications it's embedded into.

1. Introduction

Intelligent virtual agents are software components designed to act as virtual advisors into applications, especially web ones, where a high level of human computer interaction is required. Indeed their aim is to substitute the classical WYSIWYG interfaces, which are often difficult to manage by casual users, with reactive and possibly pro-active virtual *ciceros* able to understand users wishes and converse with them, find information and execute non-trivial tasks usually activated by buttons pressing and menu choices. Frequently these systems are coupled with an animated 2D/3D look-and-feel, embodying their intelligence via a face or an entire body. This way it's possible to enhance users trust into these systems simulating a face-to-face dialogue as reported in [1].

A very complete agent of this kind, frequently called an ECA (Embodied Conversational Agent), is REA [1], a Real Estate Agent able to converse with the users and sell them a house with regard to their wishes and needs. The interaction occurs in real time via sensor acquiring user facial expressions and hands pointing; moreover speech recognition is performed to bypass users need of writing their requests. REA answers using its body posture, its facial expressions and digitized sounds rendering the salesperson recommendations and utterances. The EMBASSI system [2] was born by a very big consortium occupied in defining the technologies and their ergonomics requirement to implement an intelligent shop assistant to facilitate user purchasing and information retrieval. These objectives are pursued by multi-modal interaction: common text dialogs as well as speech recognition devices are used to sense user requests whilst a 3D face is the front-end of the system. The agent is able to send its response also via classical multimedia content (video-clips, hyperlinks, etc.). In [3] an agent is described which is not just able to be animated but also to answer to users based on emotions modeled on the Five Factor Model (FFM) of personality [4] and implemented using Bayesian Belief Networks. Moreover the Alice chatterbot is used in order to let the web agent process and generate responses into the classical textual form. The remainder of this paper is organized as follows: the next section depicts the overall architecture of the SAMIR system while Section 3 gets into Animation System details. Section 4 explains the Behavior Generator module mechanics and Section 5 describes the Event Interpreter working. Section 6 addresses experimental results and finally, Section 7 draws out future work directions.

2. Overview of the System

The SAMIR system is a tool for designing a personal digital assistant where an intelligent web agent is integrated with a purely 3D humanoid, robotic, or cartoon-like layout [5]. The

architecture of SAMIR is depicted in Fig. 1, where client-side modules are box-shaped whilst server-side ones are oval-shaped.

The Event Interpreter is responsible for directing the flow of information in our system: when the user issues a request from the web site, via a common form, an HTTP request is directed to the ALICE (www.alicebot.org) servlet to obtain the HTTP response storing the chatterbot answer contained into the AIML (Artificial Intelligence Markup Language) files (i.e. the knowledge of the bot).

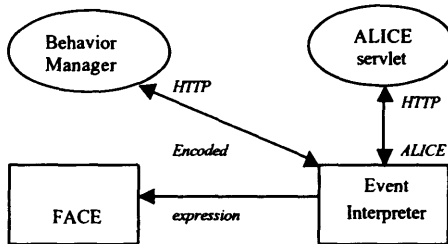


Figure 1. The overall architecture of SAMIR

At the same time, based on the events raised by the user on the web site and on his requests, a communication between the Event Interpreter and the Behavior Generator is set up. This results into a string encoding the expression the FACE System should assume. This string specifies coefficients for each of the possible morph targets [6] into our system: we use some high-level morph targets corresponding to the well-known fundamental expressions [7] but even low-level ones are a feasible choice in order to preserve full MPEG-4 compliance. After this interpretation step a key-frame interpolation is performed to animate the current facial expression.

3. The FACE Animation System

The FACE (Facial Animation Compact Engine) Animation System is an evolution of the Fanky Animation System [8] following the same philosophy but focusing more on the use of key-frame interpolation as a numerical method to animate the 3D faces used into our system. This choice was picked up in order to achieve the best performances into Shout3D API applets; moreover we can use its plug-in to export morph targets animations straight from the 3D Studio Max 4.0 modeler (www.discreet.com/products/3dsmax), considerably shortening the setup process required for integrating a new face into the SAMIR system. FACE is optimized for lightness and performance and it supports a variable number of morph targets (for example we use currently 20 high-level ones but we might use the number of the entire FAP set, in order to achieve MPEG-4 compliance) and an unlimited number of timelines. This way we can use one channel for the stimulus-response expressions, another one for eye-lid non-conscious reflexes, another one for head non-conscious reflexes and so on. In Fig. 2 some expressions taken from an animation stream are illustrated.

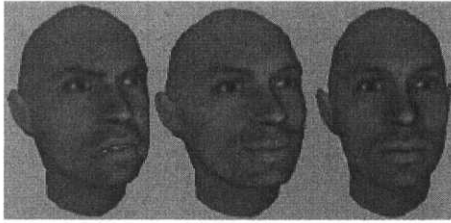


Figure 2. Some expressions assumed by a 3D face.

4. The Behavior Generator

The Behavior Generator module is devoted to generate the intelligent behavior of the agent. The Behavior Generator module aims at improving the naturalness of the dialoguing agent. The main goal of the Behavior Generator is to display suitable facial expressions according to the context of the current dialogue. It's built upon an XCS [9], an accuracy based LCS [10] classifier system, which uses a reinforcement learning algorithm, quite similar to Q-learning [11]. At discrete time intervals, the agent observes a state of the environment, takes an action, observes a new state, and, finally, receives an immediate reward. Moreover this reward gain is used by the XCS to evolve the best rules set for the tasks to achieve via a genetic algorithm. In the User Interaction module, rules are expressed in the classical form *if*<condition> *then* <action>, where <condition> (the state of the environment) represents a combination of 8 possible events, sensed by 8 effectors, such as user classification (*don't care, novice, teacher, expert*), user request to the agent, operation performed by the user in the web site, amount of user errors, while <action> represents the expression that the Animation System displays during user interaction. Specifically, each expression is represented by a linear combination of the six fundamental expressions [7] - *surprise, sadness, joy, fear, disgust* and *anger* - in which each one of the fundamental expressions is included with a percentage ranging from 0 to 100.

5. The Event Interpreter

The Event Interpreter is the module converting all the events occurring in the course of the user interaction into parameters. These events have been classified in:

- User Triggered Events (UTE), corresponding to the dialogue between the user and the system.

- Browser Triggered Events (BTE), representing user commands sent to the browser.

The Event Interpreter module extracts information to be processed by the Behavior Generator module. It's a simple applet storing the client-side state and communicating with the Behavior Generator and the Alice servlet, built modifying slightly the Alice server which was a stand-alone Java application to let it run a Tomcat 4.0 servlet (jakarta.apache.org/tomcat). Users can chat with SAMIR, because of the knowledge of the system stored into its AIML files.

6. Experimental results

Two different sets of experiments have been performed so far. In the first one, the goals were to stress the capability of the XCS component along to verify the effectiveness of the XCS when learning a set of predefined interaction rules and its ability when discovering

new rules. In the very beginning, we defined a set of 30 interaction rules that take into account a number of combinations of kinds of users (*novices, experts*, etc.) with several situations that may occur in the course of an interaction. This set of predefined rules represented the training set, the minimal know-how that SAMIR should possess to start its work in the web site.

To evaluate the performance of the system during the training phase, the following procedure has been adopted for the credit assignment:

```
If system action matches expected action then credit=100
Else credit=k/d(system action, expected action)
```

Here *k* is a constant and the distance measure *d* is defined as the Euclidean distance among the 6 components of the expression *surprise %*, *sadness %*, *joy %*, *fear %*, *disgust %*, *anger %*). The experiments aim at verifying the effectiveness of the User Interaction module to learn the 30 predefined rules. We performed 5 runs of the system with the same parameters but with different initial populations [5]. The values of the parameters have been empirically selected. On average, the system was able to learn 26 out of 30 rules. The 4 unlearned rules were, in most cases, very close to the original predefined ones. The most interesting result has been the ability of the system to discover new rules, not included in the original set of 30 rules. Due to the inherent features of XCS, SAMIR has been able to learn quite effectively the predefined rules of behavior and to generalize some new behavioral patterns that could update the initial set of rules (as in Table 1).

Table 1. Results of the training phase

Rule	Expected Output	System Output	System Condition
1	0 0 0 0 60 40	0 0 0 0 60 40	0#0#001001100
2	0 0 60 0 0 0	0 0 40 0 0 0	0001111011000
3	-	70 0 30 0 0 0	-

Another kind of experiments was performed to verify the capability of the Event Interpreter module to dialogue with users and, consequently, their reactions in front of a digital chatter. A sample of 10 users was selected, with different skills and knowledge about our system. Users were asked to interact with SAMIR (see Fig. 3) during an entire week, and to report their comments and suggestions. During the first 3 days, users were free to choose the topics of the conversation, in order to familiarize with the system. In the second part of the week, the topics of the conversation were imposed (*Computer Science, Gossip and Sport, Politics and Religion*). All the interactions have been recorded in order to analyze quantitative and qualitative factors. Over the period of the experimentation, 1,096 dialogues were recorded (an average of 100 dialogues for each user). The dialogues length ranges from 20 minutes to 293 minutes and each user spent, on average, 70 minutes per day interacting with SAMIR.

The mean length of the dialogues was 126 minutes over the free conversation period and 108 minutes over the guided conversation period.

From the analysis of the log files we noticed several situations in which SAMIR was not able to keep the dialogue going with the users. We classified these situations in 6 different categories. Table 2 shows the frequency observed in the logs with respect to each provided category. At the end of the experiment, users were asked to report their own opinions about their dialogues with SAMIR (see Table 3).

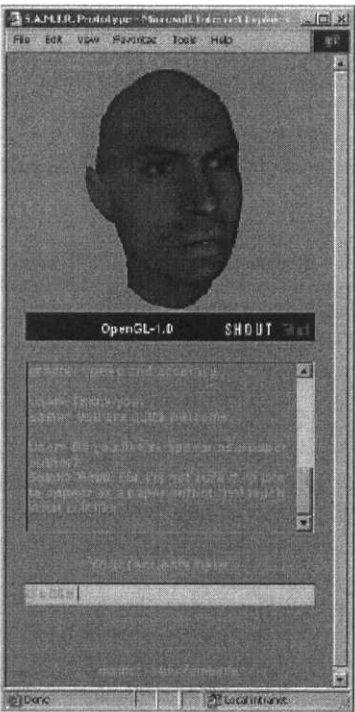


Figure 3

Table 2. Frequency of the most common errors.

Category	Frequency
Quoted Question	29%
Fall-back	20%
Misunderstood Question	16%
Wrong Answer	15%
Evasive Answer	11%
Unknown topic	6%

Table 3. User evaluation of the dialogues.

Evaluation	Frequency
Interesting	80%
Funny	80%
Ambiguous	80%
Intelligent	60%
Involving	40%
Sensible	40%
Real	30%
Up-to-date	30%

From the analysis of the separate dialogue logs, both positive and negative characteristics emerged.

- *Positive characteristics.* In some dialogues, the Event Interpreter module exhibits a sort of ironic behavior. In another situation the module has been able to defend itself from the critics of a user. In several situations the chatterbot gave some non-sense answer but, unexpectedly, users found it very nice and were attracted by it.
- *Negative characteristics.* The chatterbot is not able to manage indefinite pronouns. In very complex dialogues concerning many different interrelated topics, the chatterbot was not able to continue the dialogue. This situation occurred more frequently when dialogues were on very specific topics like *Science, Religion or Sport*.

Currently, we are employing SAMIR as a virtual advisor to users book shopping. During the first interactions, users were asked to search specific books whose author and subject was a priori fixed. On average, analysing the dialogue logs, we observed that the chatterbot found about 30% out of users required books: this negative result is due to two different reasons. The former is some users attitude to express in a very informal way, one not expected indeed. The latter is due to the inner nature of AIML: having large knowledge bases many times causes unexpected topic conflicts because of rules resolution ambiguity. We are investigating how to simplify SAMIR knowledge without sacrificing too much his conversational skills.

7. Future work

Even if we can rely a prototype performing quite well, our work will be aimed to give a more natural behavior to our agent. This can be achieved improving dialogues, and eventually, the text processing capabilities of the ALICE chatterbot, and giving the agent a full proactive behavior: the XCS should be able not only to learn new rules to generate facial expressions but also to modify dialogue rules, to suggest interesting links and to supply an effective help during the site navigation.

References

- [1] J. Cassell et al. (eds.), *Embodied Conversational Agents*. MIT Press, Cambridge, 2000.
- [2] M. Jalali-Sohi & F. Baskaya, A Multimodal Shopping Assistant for Home E-Commerce. In: *Proceedings of the 14th Int'l FLAIRS Conf.* (pp. 2-6). Key West, 2001.
- [3] S. Kshirsagar & N. Magnenat-Thalmann, Virtual Humans Personified. In: *Proceedings of the Autonomous Agents Conference (AAMAS) 2002*. Bologna, 2002.
- [4] R.R. McCrae & O. P. John, An introduction to the five factor model and its applications, *J. of Personality* **60** (1992) 175-215.
- [5] F. Abbattista et al., An Agent that Learns to Support Users of a Web Site. In: *Soft Computing and Industry – Recent Applications*. Springer-Verlag, Heidelberg, 2002.
- [6] B. Fleming & D. Dobbs, *Animating Facial Features and Expressions*. Charles River Media, Hingham, 1998.
- [7] P. Ekman, *Emotion in the human face*. Cambridge University Press, Cambridge, 1982.
- [8] F. Zambetta et al., An Agent To Personalize User Interactions. In: N. Rozic, D. Begusic (eds.), *SoftCOM 2001 – Int. Conf. on Software, Telecommunications and Computer Networks*. FESB, Split, 2001, pp. 431-438.
- [9] S. Wilson, Classifier fitness based on accuracy, *Evolutionary Computation* **3**(2) (1995) 149-175.
- [10] J.H. Holland, *Adaptation*. In: R. Rosen & F.M. Snell (eds.), *Progress in theoretical biology*. Plenum, New York, 1976.
- [11] C.J.C.H. Watkins, *Learning from delayed rewards*, PhD thesis, University of Cambridge Psychology Department, 1989.

SADISCO: A Scalable Agent Discovery and Composition Mechanism

James J. Nolan, Arun K. Sood, Robert Simon

Center for Image Analysis, Dept. of Computer Science, George Mason University, Fairfax, VA USA

Abstract. Peer-to-peer systems have recently gained popularity as a way to share files amongst distributed users. Such an approach can be applied to the discovery of distributed software agents. In this paper, we introduce a scalable agent discovery mechanism that utilizes a semantic layer on top of traditional middleware, and forms a hierarchy representing the types of agents on the network. The approach is used to support the composition of meta-agents, or "an agent of agents", to build distributed applications. Our results show that the SADISCO approach scales well and allows users to discover agents with little or no *a priori* information.

1 Introduction

Recently, distributed file-sharing systems such as Napster [9], Freenet [3], Gnutella, and Morpheus [8] have gained popularity as a way for users to easily search and share files with one another. These systems demonstrate that, when given an interface that removes the complexities of such a system, users will use it aggressively to share information. The success begs the question: can such an approach be extended to the sharing of distributed services, or more specifically, distributed software agents? At its core, a software agent according to [16] is defined as "a computer system that is *situated* in some *environment*, and that is capable of *autonomous action* in this environment in order to meet its objectives". These agents usually use some means to advertise themselves in the form of an agent description. In our previous work [14] we have presented an overall agent architecture that utilizes the Resource Description Framework (RDF) encoded in XML (what we term I-XML) as a means to advertise agents. In this architecture, when an agent enters the network, it advertises itself by placing its agent description in a repository containing all current active agents. Due to the nature of agent-based systems, it is likely that the system may grow to support thousands of agents. Thus, the problem is how to best organize the agent descriptions so that we can support scalable search. In our context, search refers to the ability to discover an agent that can assist in solving a particular query.

Napster uses a centralized index which obviously will not scale. Freenet, Gnutella, and Morpheus all utilize some notion of neighbors and a limit on how far the search query will proliferate. This property supports scalability but limits the answers to the results provided by a small subset of nodes on the network. Additionally, all of these approaches use a boolean measure of query verses data, either the query matches the dataset or it does not. In this paper, we present SADISCO, the Scalable Agent Discovery and Composition mechanism. SADISCO allows for users to submit open-ended queries (e.g., "Find me an agent that can do X?") and result in a relevancy-ranked list of results based on keyword similarity. SADISCO

supports scalability by taking advantage of two things: 1) the similarity between agent descriptions, and 2) the processing nodes on the network. When agents register on the system, a hierarchy representing the similarity between agents is formed and distributed across the network. Clients can fully search this hierarchy searching for agents for a specific task. We show the approach is scalable, testing our results with up to 100,000 agents on the network.

The remainder of this paper is as follows. Section 2 discusses related work in service/agent discovery. Section 3 presents the SADISCO architecture. Section 4 presents the results of SADISCO in our overall agent architecture. Finally, section 5 concludes.

2 Related Work

Agent discovery is important from two aspects: 1) initial discovery - the client is initially seeking a agent, and 2) discovery when an agent fails - the client needs to discover a agent because an agent the client is relying upon has failed. The mechanism to perform these two types of agent discovery are similar, however in the latter case, the client must react to a agent failure and find another agent match. The current distributed object computing (DOC) environments relies on interface or property definitions mentioned previously to discover services. As most agent-based systems are built on top of a DOC environment, we begin our comparison there. The client defines the interface name and/or property information to a centralized naming service and is answered with either references to the desired object types or a null if no matches are found after a specified time. Should a service become unusable for some reason, either through a node failure, or because the service is overloaded with requests, CORBA [10], RMI [7], and Jini [1] require the client to discover another service that implements the same interface or has similar properties as the failed service. In this section, we discuss more semantic approaches to service discovery.

The Ninja project [5] offers a secure service discovery service (SDS) [2], providing a secure directory style-access to available services. The Ninja architecture is hierarchical in nature, with trees of SDS servers connected at the top level. As news services are added, the service description, represented in XML, is assigned an SDS server then propagated throughout the SDS network to the other servers. Propagation is done in a summary, or lossy fashion, not all service information is propagated throughout the network.

The Pastry system [12] provides a scalable, distributed object location and routing substrate for wide area distributed applications. Pastry is a self-organizing network of nodes, where each node routes client requests and interacts with local instances of one or more applications. Any computer that is connected to the internet can act as a Pastry node, subject only to application-specific security policies. Each node maintains a routing table, a neighborhood set, and a leaf set. Pastry works as follows: when a message arrives at a node, the node checks its leaf set to see if the message falls within the range of its leaf nodes. If it is, the message is forwarded on to the destination. If the message is not covered by the leaf set, then the routing table is used and the message is forwarded to a node that shares a common prefix from the 128-bit node identification. The performance of this approach has been shown to be $\log_{2^b} N$ steps, where b represents the number of bits used to represent the prefix of node identifications and N represents the number of nodes on the network.

The Chord system [15] essentially provides a fast distributed mechanism of a hash mapping function that maps keys to values. This is accomplished through consistent hashing [6], a mechanism that, within a certain probability, balances the load (all nodes receive the roughly

the same number of keys) and minimizes the number of keys that must be moved when a node joins or leaves the network. Chord improves on consistent hashing by removing the requirement that every node know about every other node.

The Content-Addressable Network (CAN) [11] is an Internet-scale system that provides a distributed hash table which maps keys to values. Similar to Chord and Tapestry, CAN maintains a coordinate routing table that holds the IP address of neighbors in the routing space. When a "key" or message is submitted, it is passed through the routing space by matching the keys to values in the local hash tables.

3 SADISCO Overview

The Scalable Agent Discovery and Composition (SADISCO) mechanism provides for agent discovery utilizing two important yet simple techniques. First, queries for agents are matched using a semantic keyword matching mechanism based on the Salton SMART algorithm [13]. This approach has several useful properties. It allows for the user query to be compared to agent descriptions in a relevancy-ranked fashion. That is, a user query is compared to all agent descriptions (in their I-XML form) and results above a specified threshold are returned in a relevancy ranked list. In addition, it allows for users to submit open-ended queries to discover available agents. Second, we organize agents in a hierarchical fashion and distribute them across the network. This approach allows our search to be scalable and fully distributed utilizing all network resources. In this section we first present our semantic matching approach and then discuss how we apply it on top of a distributed hierarchy.

3.1 Semantic Matching

We have implemented Salton's vector model information retrieval algorithm [13], and we use this as the basis to match queries up with appropriate agents, who in turn make use of the algorithm to find other agents to assist in processing. The algorithm works as follows:

$$\text{sim}(a_j, q) = \frac{A_j \bullet Q}{|A_j| * |Q|}$$

where $\text{sim}(a_j, q)$ represents the similarity of agent j to query q , A represents a vector of agent descriptions, and Q represents a vector of the terms from query q . This formula states that the similarity of the capabilities of an agent a_j to a particular query q can be calculated by taking the cosine of the angle between the vectors A (the terms of the agent description) and Q (the terms of the query). The weights of the vectors A_j and Q are determined by taking the product of the normalized frequency and the inverse document frequency: $TTF/MTF * \log(N/n_i)$ where TTF is the total term frequency of a term in the agent description or query, MTF is the maximum frequency of any term in the agent description or query, N is the number of agents on the network, and n_i is the number of service descriptions in which the term i is found.

The terms available in the repository of agent descriptions is updated each time an agent enters or leaves the network. Using these terms, each agent has the capability to calculate its relevance to specific queries posed by a user, and also search for agents it may require assistance from during processing. This is especially useful as we make very fine-grain agents,

isolating very specific functionality, and these agents rely on other agents to fulfill processing in response to queries. We now discuss of how this algorithm is used in our agent-based system.

3.2 *Clients Discovering Agents*

In the Client-Agent discovery process, a user is submitting a query, by means of a client agent, for processing. This is usually some high level query such as: "What will be the impact of an outer beltway on Washington D.C.'s traffic patterns?". When the user submits this query to the system, they are seeking the best possible solution. To make this match, we measure the similarity of the query to each agent description on the network. Each agent description is text-based, and in a format as defined by our ontology.

Returned to the client is a list of agent "bids", or ranked relevance to the query. This process requires the human-in-the-loop to choose the most appropriate agent "bid" against the query. When the user selects the most appropriate agent, they then can fill in the details of the processing, including input data sets and agent parameters.

3.3 *Agents Discovering Agents*

We have defined a model to implement low-level imagery and geospatial processing agents. These agents are well described in the form of a name, input and output types, and parameters. These agents can be used indirectly to solve high-level queries. They are used by application specific agents that directly map from a user query to agent form.

These application-specific agents theoretically come in one of three forms:

1. Those that provide no additional functionality, and rely solely on atomic agents for processing.
2. Those that provide some additional functionality, and rely partially on atomic agents for processing.
3. Those that provide all of the functionality, relying on no atomic agents for processing.

We argue that our approach (agent types (1) and (2)) makes agent-development easier and quicker than approach (3). In approaches (1) and (2), agent dependency, or the ability to discover other agents that can fulfill processing, becomes very important. This is fulfilled through the use of our agent ontology, describing an agent by certain parameters as well as its *agent dependencies*.

By using a well-defined ontology, higher-level agents can now search for the atomic agents they are dependent upon by agent description. The opposite of the Client-Agent model, this search is fully automated. The high-level agent chooses a threshold, and will select the highest matching agent as its dependant agent as long as its relevance meets or exceeds the threshold.

3.4 *Creating a Scalable Search Space*

Utilizing the Salton search model provides us with a mechanism for users to submit textual queries and have them compared to agents on the network in a centralized space. A shortfall

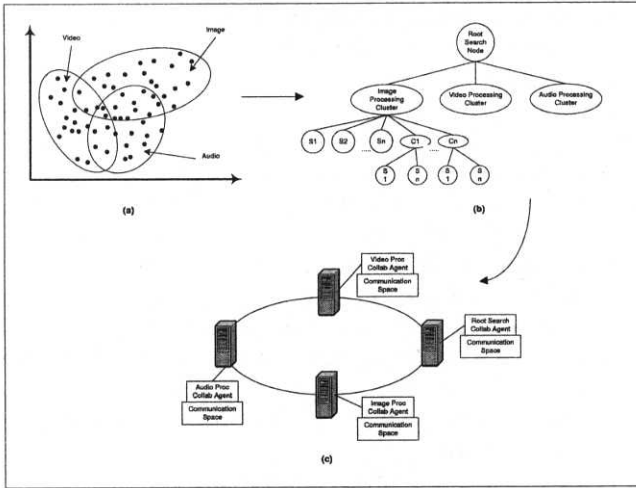


Figure 1: Creating Hierarchical Clusters in Support of Search

of such an approach is scalability, responses will be on the order of $O(n)$ where n is the number of agents available on the network. As we scale to thousands of agents, the latency of the search will become unacceptable for a user. To address this shortfall, we have adopted a hierarchical approach, one where agent descriptions are clustered based on a similarity measure and distributed across the network. Such an approach solves the scalability problem and moves us closer to an $O(\log n)$ type of performance.

The goal in this approach is to utilize peer processing nodes as resources that can assist us in answering the user query while minimizes the latency of the response to a query. This approach can be best compared to the Peer-to-Peer (P2P) file sharing mechanism that has been demonstrated in Napster [9], Gnutella [4], and more recently Morpheus [8]. An ideal view of this paradigm allows every client on the network to also act as a fully searchable server. The motivation for this is to allow distributed users to share data that resides on their local machines with one another. In reality, a true P2P approach will not scale as it allows a client to search everything in search of something. We simply cannot realistically build systems that search everything available on the network. In the simplest example, Napster, a central index captures the data holdings of all clients. When a client submits a query, the index is searched and others who can provide the data are returned. While very efficient and providing a true global search, this approach is a central point of failure and a target for malicious attacks. Gnutella moves closer to a true P2P system as the indices are distributed across clients. Searches are based on a "horizon", or how far the client can extend to other clients, and a Time To Live (TTL), or how far the query should be propagated. This approach eliminates the single point of attack, distributes the query across nodes, however the results of a query are highly biased by the client's neighbors, the effect being a local search. Morpheus improves on the Gnutella approach by creating clusters of peers, that represent clients on the network. When a client submits a query, it can be exchanged among super-peers, improving the extent of the search while also improving performance. Because this approach relies on the communication between super-peers, its performance is highly dependant on the

construction and communication network between them.

To develop a this hierarchy, we consider several things: the types (or classes) of agents available on the network, the available locations (or processing locations) and the network, and the depth of the hierarchy. A concrete example illustrating the construction of the hierarchy is shown in Figure 1. In Figure 1a, we show that the agents available on the network are clustered based on the type of work they perform. In this case, we show 3 clusters: agents that perform image processing functions, agents that perform audio processing functions, and agents that perform video processing functions. It is important to note that these clusters are not necessarily linearly separable. That is, we may have two clusters that refer to the same agent. The effect of this is that a query may result in multiple agents that can assist in its solution. Next, as shown in Figure 1b, a hierarchy is formed by going deeper into these clusters, creating more refined ones until finally the agents themselves are located at the leaves of the hierarchy. Finally, this hierarchy is mapped onto the network, with the root node fully replicated at each node. This replication provides each client with the capability to fully search the network for a particular service.

This approach moves us to a scalable search space that will support thousands of agents on the network as we will demonstrate later. When a user submits a query, it enters at the fully replicated root node of the hierarchy. This query is compared with all of the clusters at the next level of the hierarchy and passed on to the node with the highest similarity, or if the user requires a less precise answer, to any nodes within a certain threshold of similarity. This process continues until the query gets to the leaf nodes of the hierarchy, where the most similar services are packaged up and returned. It is important to note that our approach assumes that the hierarchy is fully distributed except for the root node which is fully replicated. For that reason, network delay in the response time must also be incorporated. Additionally, creating the hierarchy presents several issues that are outside the scope of this paper including: determining the optimal hierarchy, distributing the hierarchy across the network, handling failed nodes, and the drifting of cluster centroids as agents fail and new agents register on the network.

3.5 Using Discovery to Support Composition

With the the fully distributed, semantic agent discovery mechanism now presented, let us briefly discuss how this is used in agent composition. Our approach is to build a set of "atomic" agents, agents that do simple processing, and deploy them on the network. With descriptions available on the network and fully searchable, this allows a user to search for agents and "compose" them into a new meta-agent, or "agent of agents". These agents are fully communicative and can be represented as a graph. A screenshot of such a composition can be seen in Figure 2.

Here the user can search for agents, select agents from the network, link them together, edit their parameters, and deploy them as a new, meta-agent. In the upper left quadrant of the screenshot is the Agent List Panel. This panel gives a list of all agents available on the network, or a user can search for a specific agent by typing in a textual description. In the lower left quadrant is the Knowledge List Panel. This panel allows the user to search the knowledge repository and see how and what agents have been used to solve particular queries. Users can view the agent composition, even down to the detail of the parameters and their setting for each agent composition. In the upper right quadrant of the screenshot is the Agent

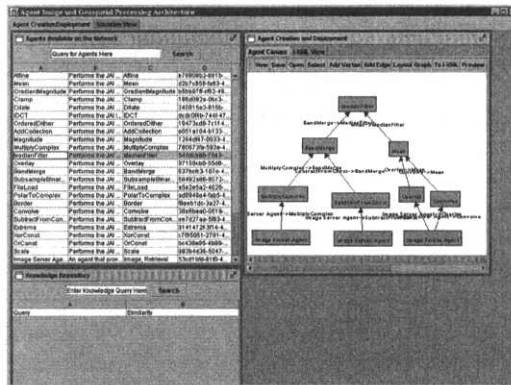


Figure 2: Agent Discovery, Composition, and Deployment

Canvas. From the Agent List Panel, users can drag agents onto this canvas, link them together, and create a new meta-agent for processing.

4 Results

In this section, we present two sets of performance results. First, we present the results of our semantic search mechanism (based on the Salton algorithm) as a layer on top of traditional middleware. We present our results as a layer on top of the Jini Lookup and Discovery mechanism, and we expect results would be very similar with the CORBA Trader service. Second, we present the results on our fully distributed, hierarchical approach. This approach extends the first by decentralizing the index of available agents and forming a hierarchy across the network.

4.1 Semantic Layer Results

In our agent testbed, we rely on Jini to provide the communication infrastructure and have developed our service discovery layer on top of it. Our approach demonstrates the ease of adding the information retrieval layer on top of existing middleware. To test our service discovery mechanism, we developed an experiment searching for services using our information retrieval approach and compared them to Jini. In the semantic case (referred to as I-XML in the below figures), we formulated a query that we knew would result in positive results: "Is there an agent that can perform the Add function?". There is no direct way to convert this into a Jini discovery template, so we manually mapped the query into a Jini Entry that would guarantee the same set of results. In this mapping, we created public fields of type String that encompassed the I-XML description of a service, this included "name", "description" and "keywords" (array of Strings) fields. These fields were populated with the appropriate values. We then submitted the query to the I-XML Collaboration Agent and the Entry to the Jini Lookup service and measured results.

Our prototype system currently has 100 unique imagery, video, textual, and geospatial processing agents. We tested up to 500 agents by permuting these original 100 to guarantee

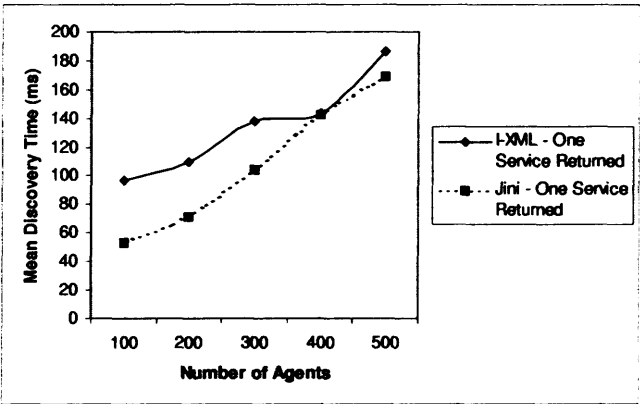


Figure 3: Mean time to discover one agent as compared to the number of agents on the network

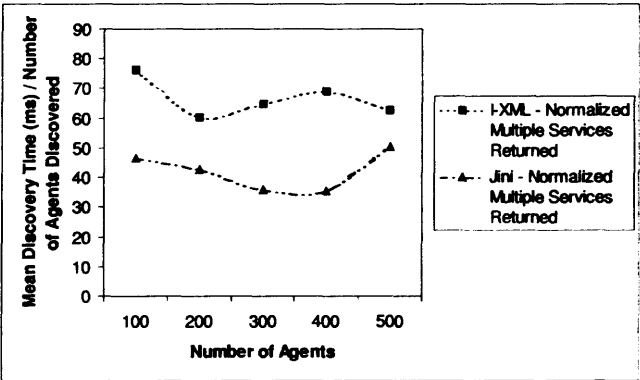


Figure 4: Normalized time to discover multiple agents as compared to the number of agents on the network

a new set of 100 unique agents. At each increment of 100, we tested the time to discover one service and multiple services. Each increment of additional services added another service similar to the known service, so at 200 agents, 2 services were returned, 300, 3 services, etc. We ran 1000 queries each time, 10 iterations of 100, and took the average round trip time to return an answer, or service(s) that can assist in answering the query.

The results of our experiments can be seen in Figures 3 and 4. We note that as the number of services returned increases, the difference in discovery time between I-XML and Jini increases. On the other hand, Figure 4 shows the difference between the normalized times for I-XML and Jini are constant. These results show that our approach is comparable to traditional service discovery techniques with the added benefit of unstructured query processing. Since our service discovery layer is built on top of the middleware layer, these curves show the additional overhead incurred to provide the capability for service discovery under partial specification. In other words, since we rely on the Jini Lookup Service or the CORBA Trader

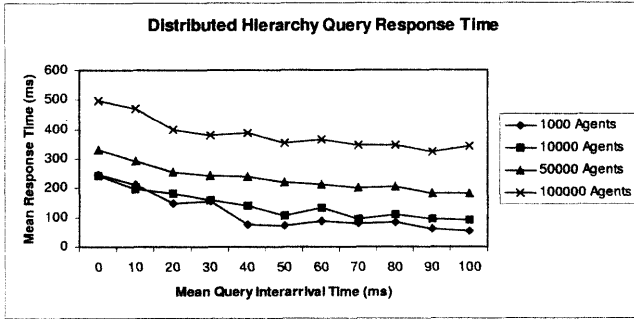


Figure 5: Hierarchical Service Discovery Results

Service as the basis for our discovery mechanism, our approach will always add a certain amount of overhead to those approaches. However, the benefit for that small cost is a search approach that provides higher resolving power than traditional middleware.

4.2 Distributed Hierarchy Results

The semantic layer previously described is centralized, it resides on top of a middleware discovery mechanism such as the Jini Lookup Service or the CORBA Trader service. As the results show, this approach works well with a small number of agents, but as we scale up to thousands or even hundreds of thousands of agents one can see that the response time to discover one agent will soon become unacceptable from the user perspective.

To address this issue, we have developed a hierarchical approach as discussed in Section 3.4. We tested this approach in our lab on a network of six Sun Solaris Ultra 10s. We developed a hierarchy with three levels. At level one resided the master node, level two contained 12 clusters, and level three contained 74 clusters. Hanging from level three were the services themselves (leaf nodes). The master node is the entry point for all queries and was fully replicated at each node. We simulated one thousand, ten thousand, fifty thousand, and one-hundred thousand agents on the network. Once the hierarchy was created, the simulated service descriptions were deployed and tested.

We simulated query arrivals as a Poisson process using an exponential distribution and adjusting the mean interarrival time. We tested mean interarrival times of 20-100 ms in 10 ms increments and tested one thousand queries, each with an independent arrival time. The results of our approach can be seen in Figure 5. These results are not meant to be absolute measure, however they show the relative scalability of our approach and will depend on the processor types used in the hierarchy, network bandwidth, and other utilization of the system. We point out that these results were obtained on a shared network in the Computer Science department, they include contention with other users that we can neither control nor measure.

As expected, response times begin to degrade as the mean interarrival time is decreased and queues are starting to build up throughout the hierarchy. What these relative results do show however, is the scalability of our approach. A ten fold increase (from 1000 to 10000 agents) results in a fractional delay in response, and increasing by another factor of ten adds a larger but still acceptable amount of delay.

5 Summary and Conclusion

We have presented SADISCO, the Scalable Agent Discovery and Composition mechanism which adds a semantic layer on top of traditional middleware allowing for unstructured queries. We have extended this semantic approach to form a hierarchical, Peer to Peer (P2P) approach for agent discovery. This approach clusters agent descriptions based on their similarity, and distributes nodes of the hierarchy across the network. We simulated up to 100,000 agents on the network and have demonstrated this approach to scale well by providing acceptable round-trip query response times.

References

- [1] Ken Arnold, Brian O'Sullivan, Robert W. Scheifler, Jim Waldo, and Ann Wollrath. *The Jini(TM) Specification*. Addison-Wesley, Reading, MA, 1999.
- [2] S. Czerwinski, B. Zhao, T. Hodes, A. Joseph, and R. Katz. An architecture for a secure service discovery service. In *Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM '99)*, Seattle, WA., August 1999.
- [3] Freenet. <http://freenet.sourceforge.net>.
- [4] Gnutella. <http://gnutella.wego.com>.
- [5] Steven D. Gribble, Matt Welsh, Rob von Behren, Eric A. Brewer, David Culler, N. Borisov, S. Czerwinski, R. Gummadi, J. Hill, A. Joseph, R.H. Katz, Z.M. Mao, S. Ross, and B. Zhao. The ninja architecture for robust internet-scale systems and services. *Journal of Computer Networks*, 35(4), March 2001.
- [6] D. Karger, E. Lehman, F. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 654–663, El Paso, Texas, May 1997.
- [7] Sun Microsystems. Java remote method invocation specification, version 1.7. Technical report, Sun Microsystems, December 1999.
- [8] Morpheus. <http://www.musiccity.com>.
- [9] Napster. <http://www.napster.com>.
- [10] OMG. The common object request broker: Architecture and specification, version 2.4.2. Technical report, The Object Management Group, February 2001.
- [11] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. In *Proceedings of the ACM SIGCOMM*, San Diego, California, August 2001.
- [12] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location and routing for large scale peer-to-peer systems. In *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001)*, Heidelberg, Germany, November 2001.
- [13] G. Salton. *The SMART Retrieval System*. Prentice Hall, Englewood Cliffs, NJ, 1971.
- [14] Robert Simon, James J. Nolan, and Arun K. Sood. A light-weight agent approach for collaborative multimedia systems. *Information Sciences*, 140(1/2):53–84, January 2002.
- [15] Ion Stoica, Robert Morris, David R. Karger, M. Frans Kaashock, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup protocol for internet applications. In *Proceedings of the ACM SIGCOMM*, pages 149–160, San Diego, California, August 2001.
- [16] Michael Wooldridge and Nicholas Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.

MAYBE - Multi-Agent Yield-Based Engineering : Improve Training in the Emergency Room Chain

Sophie GOUARDERES, Guy GOUARDERES, Philippe DELPY
LIUPPA, Université de Pau et des Pays de l'Adour - BP1155¹
64013 - Pau cedex - France
sophie.gouarderes@univ-pau.fr

Abstract : This paper describes a method of multi-agent analysis and design for reactive, real-time information systems, relating to complex and risks applications in medicine. According to specific needs in emergency healthcare units : spatio-temporal deployment of heterogeneous tasks, non-determinism of actors and self-organization in an unpredictable and/or disrupted environment, we propose MAYBE - Multi-Agent Yield-Based Engineering. MAYBE is a solution that makes possible for the agents to evolve and adapt by instantiation in different contexts. This paper details the various stages of the methodology applied to an emergency case, in parallel with the computerization of the process. It also compares the issues with other current work.

1. Introduction

Distributed and cognitive multi-agent systems are used in various fields, notably that of Health Care, as an aid to decision making for the different steps to train emergency staffs to the risk prevention process¹. In these systems, four orders of difficulty are encountered : the modeling itself, the syntax and conventions used for representation, the calculation aspect of the resolution, the traceability of the training session and validation of the results.

Until now, effective, user-friendly and versatile models of interactions between human agents and artifact entities have only partially integrated formal approaches such as planning, scheduling and constraint satisfaction throughout the development, utilization and maintenance cycle of intelligent interfaces [1].

Current so-called "intelligent" multi-agent systems are evolving towards a more self-organized, collective intelligence in terms of autonomous and deliberative agents [2].

This paper proposes a complete methodological approach that responds to these criteria for the design and implementation of more "cognitive" Multi-Agent Systems, more capable of adapting to emerging crisis situations (in terms of resources) or individual and / or collective dysfunctions. We have already had the opportunity to evaluate the effectiveness of flexible and cognitive MAS in the fields of aeronautics [3] and e-learning [4]. Likewise, the premises of the method presented here were drawn from these studies [5].

One question to be resolved is whether we can develop software for long life training in the hospital context, based on actors of the type described above, which is both highly reliable and sufficiently versatile to adapt to situations of dysfunction or even crisis.

¹ To : 1-Identify, 2-Understand, 3-Estimate, 4-Prevent, 5-Manage/Master/ Back up

2. Problematics

Multi-Agent Systems are a highly appropriate tool for the real-time management of the constantly modified activities of a complex process, such as the emergency room chain. In this case, they are an indispensable information system, within the global clinical situation, for studying competition, cooperation, and conflict resolution phenomena in highly unpredictable, high-risk situations for training.

Different crisis situation scenarios have highlighted the following needs :

- distribution : the tasks, resources, organization and know-how are distributed among human actors, services, jobs etc. Communication is therefore transmitted via various media and handles different ontologies ;
- competition : at any moment tasks may be carried out in parallel, generating all kinds of conflicts ;
- non-determinism : the patient, the care provider, etc. are living beings who produce unexpected events ;
- self-organization : in an unpredictable and/or disrupted world, the system's response must be flexible ; even if the protocols are strictly regulated, many decisions must be negotiated as a result of limited resources
- intensive communication : everything must be dispatched to the right place and the right person with maximum efficiency.

For these reasons, the software agents designed using MAYBE are based on the concepts of flexible determinism, rational agents and the emergence of flexible cooperation models.

In the evolution that we propose, we have the same concepts of goals, roles, activities, tasks, and models of communication and coordination. We introduce two new paradigms that are of great importance for more open environments : a. delayed assignment of goals, roles, and tasks in order to allow them to be achieved ; b. needs analysis, based on observation of the progressive constraint satisfaction steps that occur in the real process (including the time factor).

Needs are expressed as “naturally” as possible in speech acts, based on the expressions of the actors so that even they are active in the process. Any modification to a situation, (change of actor, discourse, context, etc.) is indexed by a trigger (a signal triggered by an event). Each “speech act” is situated with respect to triggers that dynamically punctuate the process in rhythm with the performatives. Each end of sequence specifies a goal attained through the sub-goals of the activities that make it up.

At this stage, we should progressively start to declare the primitives used to model the steps in a functional environment.

Next, each transaction is formally specified in terms of roles and the interactions between roles using task grammars to satisfy the internal consistency and completeness constraints. Each role, provided with a goal and a planned set of tasks, now prefigures something that will become an agent.

But this formal specification is not rigid ; each task can at any time be rewritten in the declared functional environment (like the communication model (multithread) and the coordination model (logical programming)). This characteristic will make it possible to cause the agents to evolve and adapt by instantiation in different contexts [5]. We call this approach MAYBE - Multi-Agent Yield-Based Engineering.

3. The Agent Design Methods

The two paradigms described above (emergent modeling and delayed autonomous distribution) modify the methods for designing and implementing agents. To explain the

evolution required in the agent design methods, we decided to place our approach in relation to some well-known frames of reference (cf. figure 1) : a. Analysing Agent Interaction [6], b. GAIA [7] and c. MASE [8]. These three methods support high-level abstraction (design meta-entities) in a descending approach that allows for evolutive, step-by-step structuring to facilitate the re-use and flexible maintenance of the agents.

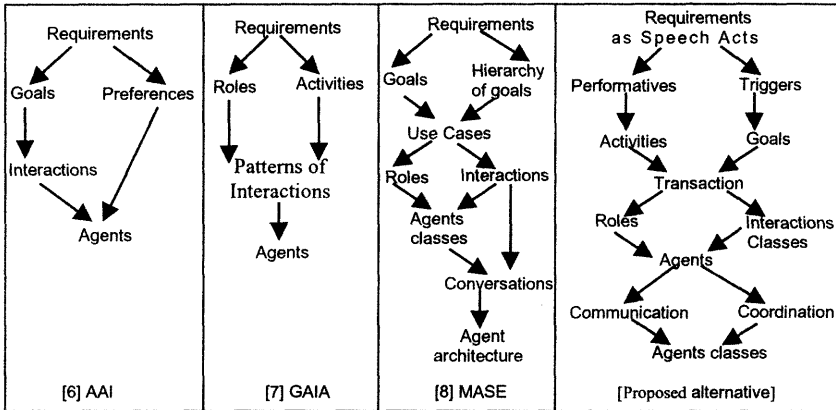


Figure 1. Respective approaches of the AAI, GAIA, MASE and MAYBE methods

The AAI approach as is true in the majority of cases, involves extracting the goals from the needs analysis. Preference analysis makes it possible to specify what the agents are, how they coordinate together, how they satisfy the accomplishment of goals, and how to justify the taking into account of preferences during the design phase.

In GAIA, it is the individual perspective of the role that prevails almost antinomically. The interactions are constructed on the basis of the roles, and for each role the activities put into play interact with the activities of the other roles. Patterns are therefore needed for these interactions, and the final phase consists in providing a range of protocols that can be used by means of tendering .

If we examine MASE, a more integrated component assembly approach, we return to the same poorly resolved problem of extracting goals from needs. Once goals are found, hierarchies can be constructed (they might be ontologies). Then the whole structure is split up into “use cases,” in order to define in each case the role and responsibility of the activities of each agent. After specifying each class of agents, the possible “conversations” (or dialogues) between them are analyzed using a state diagram (RP) in order to validate the acquaintances and the achievement of goals.

In the evolution that we propose, we have the same concepts of goals, roles, activities, tasks, and models of communication and coordination. We introduce two new paradigms that are of great importance for more open environments : a. delayed assignment of goals, roles, and tasks in order to allow them to be achieved ; b. needs analysis, based on observation of the progressive constraint satisfaction steps that occur in the real process (including the time factor).

3.1 From Dialogue Acts to Transactions between Agents

To model our method, we used speech acts. The fundamental idea in speech act theory is that a speech act can be considered to be a combination of acts or locutionary, illocutionary and perlocutionary forces. For “performative” utterances, in addition to the locutionary act,

speech act theory also enumerates acts that are performed in speaking (illocutionary acts) and as a result of what is said (perlocutionary acts). Austin classifies these “performative” acts into five categories : verdictives (judgment), exercitives (sanctioning a behavior), commissives (commitment), behabitives (congratulations), expositives (clarifications). These primitives become more operational in K. Allan’s vocabulary [9] : statements, expressives, invitationals, authoratitives.

For the example illustrating the reception of a patient into the emergency room chain in a hospital [10], we have used the simplified approach proposed by K. Allan [9] :

In this emergency room chain in a hospital, at a given moment, a "dialogue" takes place between the Reception-Orientation Nurse (RON) (r) and the patient P (p) to assign him or her a place for care : a bed or an examination room.

According to Ferber and others, the RON and P roles, attributed to the actors a priori (the ARG solution : Actor-Role-Goal), makes it possible to define two agents : RON and P whose contact will result in RON assigning P to $Bed(i)$ at instant t_i . If the contact takes place at instant $t_i + \Delta t$, then this will not be $Bed(i)$ but $Bed(j)$, such that it satisfies both the goals of RON, P , and the service (system).

In reality, if the RON is already assigned to a sick patient, a doctor D (r), who is still available may assign the patient to a bed. In the ARG system, the roles of the D and the RON must be exchanged ; at the same time he or she assigns the patient to a bed, the doctor begins the diagnosis, or even the medical care ! This case poses the tricky problem of "merging agents", because there are so many actors and even more patients, and time and space are not extendable from one end of the emergency room chain to the other.

We therefore chose to consider that the assignment of patient P is the result of a meeting between actors, actions or acts in the real world. This meeting leads to a change of context that can be detected in space and time, and that can be identified by a dialogue to which a goal is assigned, which will be attained by a transaction between agents.

To define these agents, an analysis of the different activities must be performed, not based on the roles attributed a priori to the actors in the real world, but based on the internal and external events that punctuate the process in terms of "dialogues" between these actors. This dialogue is represented by a speech act (cf. table 1).

Table 1. Example of a speech act

Receive, assertion and inter-actors (Allan 1998 -99)	
Description	Patient received as p
Pre-condition	Identification : Receiver r accepts p as patient
Elocutive Intention	Reflexive dialog confirms r as receiver and p as patient in C_p context

3.2 The Model

The method we propose consists of 7 steps (cf. figure 2), presented in the diagram below. These steps have been formalized in Scheme in order to automate the method.

The first step (Step 0) is manual and consists in analyzing the dialogues of the process to be studied in order to bring out all the triggers, activities and goals. It is based on interviews that are interpreted as dialogue acts between real actors. This progressively enables the performatives and their triggers to be extracted from an analysis of the interactions between the actors (human or artifactual). From this highly abstract division, the following steps use a succession of appropriate formalisms (GOMS, ETAG) to select the goals and define the

tasks, integrating the spatio-temporal constraints from an individual and a collective point of view.

Two models of coordination and communication defines, the acquaintancy rules and their evolutions in function of the context. The following steps in the method structure the architecture by refinement and coordination according to various models and methods (formal or semi- formal grammars). These steps are explained in detail later in this paper.

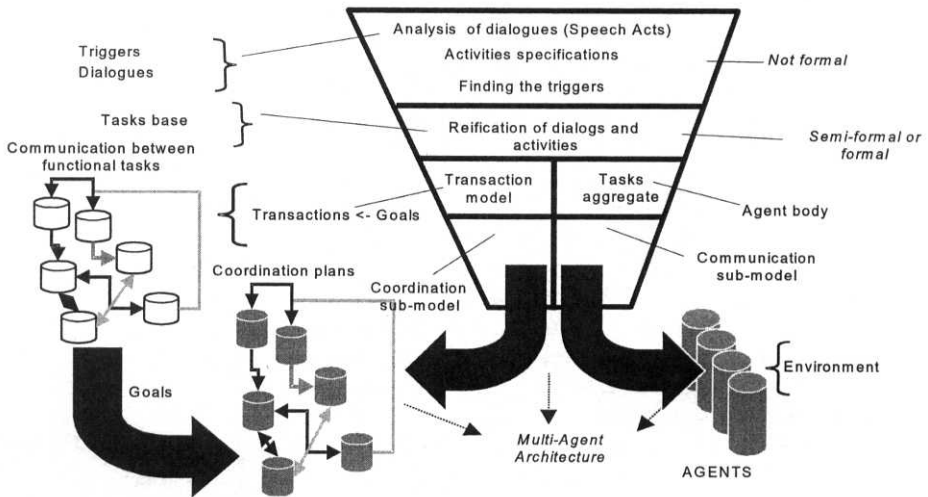


Figure 2. General synopsis of the approach

4. The Case Study : Reception of a Patient into an Emergency Room Chain

We have chosen to give a full example of our method. This example deals with the reception of a patient into the emergency room chain of a hospital. It takes up the different steps in our method, using existing models and formalisms from the scientific community.

The admission process can be described according to the following series of tasks :

- Arrival of the patient ;
- Collection of administrative information for identifying the patient, starting his/her administrative file, checking and finding out his/ her rights : third-party payment, CMU (French Universal Health Coverage), VITALE card (French health care smart card) etc;
- Making up and printing name labels – sometimes including a barcode – to identify the documents in the administrative file, examination slips and biological samples, and to make the rendering of results secure – and perhaps even putting them on-line so that the file is available in real time;
- Collection of medical information by the reception nurse ;
- Making up of administrative file ;
- Finding and/or waiting for an available examination room or bed ;
- Placing the patient in this bed and medical examination.

5. A Formalization of the Steps for Automating MAYBE

We shall now return to the 7 steps in the MAYBE method. These steps have been formalized in Scheme. In this paper, we shall limit ourselves to providing an example result for each step in order to improve readability.

5.1 Step 1 & 2: Definition of the Environment in Scheme and next Finding the Triggers

The starting point is to account for the exchange between the components of the system, using the dialogue acts approach, which provides the dialogues and triggers. In the first step, based on speech acts, we pick-up parts of dialogue rather than goals that could lead directly to the agents. But the frame of dialogues obtained is continuous and does not include any notion of time. Finding the triggers provides the timing of the illocutionary acts and makes it possible to classify them using the temporal references given by Allan. The system of triggers thus sequences the speech acts.

In order to find the triggers, we defined a function called `frame_trigger`. This function takes as its parameters the verb resulting from the speech acts, the concepts (in other words the whole set of activities, constraints, and temporal contexts), the axioms that enable the concepts to be linked two by two, and the rules that make it possible to obtain an ordered list of activities. This function enables a specific environment to be associated with a verb resulting from the speech acts.

The expected result is to be able to associate a list of activities, ordered in function of time and constraints, to a trigger resulting from the speech acts.

5.2 Step 3 : Finding the Goals

From the trigger just described in the previous step, we attempt to find its goal by means of its activities. In fact, the name of the trigger has become a parameter. With the help of the previous frame, we obtain the activities of the trigger in question as an ordered list. Each activity is connected with a goal. The goal we are looking for is the goal of the last activity, i.e. the final goal of the act under consideration.

Example for the arrival of a patient (in 3 steps) : The goal is to describe the real world in a correct and comprehensible way. In fact, from the dialogue between the different actors, we manually identify a set of triggers (T) that are necessary for using the system. After identifying the triggers, we draw up a list of the goals (G) associated with each trigger. The last step consists in constructing a set of activities (A) for each trigger and associated goal.

To find the goals, we opted for a matrix representation, where the columns represent the activities A_i of the dialogue, which may or may not be set off by a trigger, and where each line entry of the matrix (corresponding to trigger T_i) is associated with a goal " G_k " by an activity vector $\langle A_1, \dots, A_m \rangle$ of this matrix.

So we can establish the matrix (cf. table 2):

Table 2. Example of matrix to find goals.

	A1	A2	A3	A4	A5	A6	Goals
T1	1	1	0	0	0	0	G1
T2	0	1	0	0	0	0	G1
T3	0	0	1	0	0	0	G2
T4	0	0	0	1	1	1	G3

5.3 Step 4 : Reification of the Dialogues and Activities

In this step, we "reify" the dialogues and activities identified above into sequences of tasks. This step is necessary for finding the articulations between the activities in order to avoid modeling too many agents. Without this step, we would have one agent per speech act, which would lead to rigid assignment of roles to agents and, in the long run, to conflict resolution problems.

This step takes place in two stages : a semi-formal specification of activities by means of a task analysis method, which makes it possible to describe the activities resulting from the dialogues as a sequential set of tasks, and a formal specification of the methods by means of a task grammar (<events, tasks> grammar) which makes it possible to describe the graph of tasks by class of triggers and for each class of entry points in the graph of tasks (functional and formal reorganization of tasks).

The goal of the first part of this step is to obtain a finer decomposition into tasks of the activities associated with a trigger resulting from the speech acts. As for the second part, the result obtained is a detailed and formal description of each method (task) associated with the events that triggered the task. The formalization of the tasks is thus reduced by one level of granularity.

We have chosen an example using GOMS [11] for the semi-formal specification and ETAG [12] for the formal specification.

5.3.1 GOMS Specification

Once the goals have been extracted, the GOMS method enables these goals to be formalized by means of scripts and scenarios. The objective is to describe the sequence of steps required to attain the goal. Each vector of the first matrix provides a scenario.

When a patient is admitted, procedures vary depending on whether or not the patient is known to the emergency room service, which gives us the following selection rule :

```
SR1 : Selection rule to accomplish goal "admission of a patient"
      If new patient then accomplish goal "admission of a new patient"
      If old patient then accomplish goal "admission of an old patient"
      Return with goal accomplished
```

According to these rule, we found some script :

```
S1 : method for goal "admission of a new patient"
    step 1 : social allowances (M1)
    step 2 : visual allowances (M2)
    step 3 : method for goal : "create file" (M3)
    step 4 : box attribution (M4)
    step 5 : validation (M5)
    step 6 : return goal accomplished (M6)
S2 : method for goal "admission of an old patient"
    step 1 : social allowances (M1)
    step 2 : visual allowances (M2)
    step 3 : display old file (M7)
    step 4 : modify file date if necessary (M8)
    step 5 : box attribution (M4)
    step 6 : validation (M5)
    step 7 : return goal accomplished (M6)
S3 : method for goal "create file"
    step 1 : label attribution (M9)
    step 2 : whereabouts keyboard (M10)
    step 3 : read letter of the doctor (if it exists) (M11)
    step 4 : usual doctor keyboard (M12)
```

step 5 : validation (M5)
 step 6 : return goal accomplished (M6)

This step is also formalized by means of a matrix. The lines represent the set of methods (M) required to accomplish the activities. The columns represent the set of scenarios. Each vector is associated with a selection rule (SR).

So we can establish the matrix (cf. table 3) :

Table 3. Example of matrix for GOMS specification.

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	SR
S1	1	1	1	1	1	1	0	0	0	0	0	0	SR1
S2	1	1	0	1	1	1	1	1	0	0	0	0	SR1
S3	0	0	0	0	1	1	0	0	1	1	1	1	

5.3.2 ETAG Specification

The objective is, on the one hand, to construct the set of tasks "Entries" required to execute the methods of each step and, on the other hand, to extract the set of resources (*objects, states, events*). These tasks and resources are defined according to the ETAG grammar and provide the details of each method in the second matrix.

The following example only models one part of the final matrix for the M3 and M4 methods.

In ETAG, we have :

```
Type (object = box )
Value set : occupied | free
End object

Type (object = patient)
Value set : new | old
End object

Entry 1 :
(task > file creation), (event > patient), (object > patient = *P),
(object > file = *F)
    t1(event > new patient), (object > patient = p), (object file = f)
    "create the medical file of a new patient"

Entry 2 :
(task > box attribution), (event > patient), (object > patient = *P),
(object > box = *B)
    t2(event > patient), (object > patient = p), (object > box =
"free")
    "attribute a box to a patient"
```

This step is also formalized by means of a matrix. The lines represent the methods and the columns describe the objects, states, and events of the tasks. So we can establish the matrix (cf. table 4) :

Table 4. Example of matrix for ETAG specification.

	Object	States	Events	Entry : task
M3	< file >	< >	(event > patient)	< Entry 1 >
M4	< box >	< occupied, free>	(event > patient)	< Entry 2 >

5.4 Step 5 : Finding the Transactions

The design of these interaction classes could be decomposed into two phases :

1) Top-Down Analysis : reasoning on the basis of the <states, events> component of M under the constraint of planning the tasks using Colored Petri Nets, for instance [7]

2) Bottom-up Analysis : reasoning only on the basis of the Entries obtained under the constraint of the <states, events> transitions associated by selecting representative examples : for instance, a Galois lattice would be used as the indexing structure according to the principles used in IGLUE [1].

In the MAYBE method, the associations between Entries and <states, events> are already placed in the matrix (Step 4) based on the GOMS/ETAG articulation and, in addition, this matrix directly provides the selection rules for individuals. In fact, the expected results have already been attained through the complex and costly combination in the previous approach.

5.5 Step 6 : Coordination and Communication Models

The representation of interactions between agents (human or artificial) is a central problem in the structuring of multi-agent architectures. These two models are a necessary requirement in any multi-agent system for coordinating the agents. The communication model represents the synchronous and asynchronous relationships between the agents (model situating the agent in its environment), whereas the coordination model is essential for avoiding conflicts between the agents involved in a transaction (model situating the agent in a transaction).

5.5.1 Coordination Sub-model

The problem consists in sharing the allocation of resources in accordance with the "stream" mode of the STROBE (STream Object Environment) model [13]. For this model, we define a set of Scheme predicates of the following type : *use?*, *product?*, *produce?*, *is_used?*, *is_shared?*. The application on which we tested the method (the emergency service of the hospital) did not require a sophisticated real-time coordination model, because the procedures are defined by the hospital's protocols.

5.5.2 Communication Sub-model

We use the STROBE model to list the communication protocols between the different entities "1-to-1," and thus add a collective dimension to each agent. This model checks that the communication between two potential agents A and B, as defined in step 5, takes place according to the REPL (Read, Evaluate, Print, Listen) cycle in Stream mode

For example, one can note that the assignment of a variable (allocation of a bed) that is private to B, but accessible to both A and C may cause either A or C to perceive B as behaving unexpectedly (or incorrectly) when, for example, isolation rules are broken due to adverse events.

5.6 Step 7 : Conflict Resolution

The above remarks concerning adverse events do not imply that cooperation patterns are wrong or not fully completed, but that this open scheme for communicating between agents substantially shifts the focus of any realistic client-server scenario to a double integration : the Human in the loop, and different theoretical, experimental and applicative approaches are considered to be complementary resources for solving complex problems in a usable way.

Unfortunately, clinical performance may deviate from the ideal and, even if all physicians involved in acute or emergency care can be expected to perform practical procedures, we need to differentiate active errors (which are those that immediately precede an adverse event) from, for example, latent errors (which are factors inherent in a system). For this reason, we use a set of predicates (*use?*, *is_used?*, *product?*, *produce?*, *is_shared?*) as defined above for the coordination model in step 6. Nevertheless, this approach may be insufficient for a field that has weaker or more informal procedures.

When these 7 steps are completed, the final multi-agent system is made up of the library of agents validated in step 6, in accordance with the architecture derived from the coordination model in step 5.

6. Conclusion

We consider the essential result of the MAYBE method to be the interactive concurrent bottom-up analyses, based on multiple points of view between different agents. Our approach offers flexibility, which stems from the fact that agents can be modified and are autonomous, in contrast with other more integrated and rigid methodologies, based on agents defined in advance in terms of their roles.

MAYBE uses a series of models in "cascade" to go from an abstract representation of the problems to a formal one of the directly programmable agent. But, these various models lead to a real complexity of the method. This is why we wanted to automate the process. So, the used formalisms are transparent for the user. The method's use is easy and the complexity of the models is hidden as a black box for the user.

In term of implementation, a first prototype (a simulator) was developed to test the method within the emergency room chain of the "centre hospitalier côte basque", situated in Bayonne. We used an Agent Building Environment (ABE) of IBM but it was too poor in terms of inference and adaptation according to rules. So an implementation of the MAYBE framework in Scheme using Jaskemal multi-agent platform is currently in progress [14].

7. References

- [1] M. Burstein, G. Ferguson and J. Allen Integrating, Agent-based Mixed-initiative Control with an Existing Multi-agent Planning System, in *the Proceedings of the 2000 International Conference on Multi-agent Systems (ICMAS)* (July 2000).
- [2] D. Kinny, M. Georgeff and A. Rao, A methodology and modelling technique for systems of BDI agents. in :W. van de Velve and J. W. Perram eds, *Agent Breaking Away : proceeding of the seventh european workshop on modelling autonomous agents in a multi-agent world*, Lecture Notes in Artificial Intelligence #1038, (Springer-Verlag, Berlin, Germany, 1996) 56-71.
- [3] G. Gouardères, A. Minko and L. Richard, Cooperative Agents to Track Learner's Cognitive Gaps, actes du congrès *5th International Conference on Intelligent Tutoring Systems'2000* (Montréal, Canada), Lecture Notes in Computer Sciences #1839,(Springer ed, 2000), 443-453.

- [4] C. Frasson, L. Martin, G. Gouardères and E. Aimeur, LANCA : a distance Learning Architecture based on Networked Cognitive Agents. *4^o International Conference on Intelligent Tutoring Systems -ITS'98-* (San Antonio, USA, 1998), Lecture Notes in Computer Sciences 1452, (Springer Verlag) 142-151.
- [5] H. Kriaa and G. Gouardères, Revisable Analysis and Design by Actors Interaction : Emergency Case Study, in R. Kowalczyk, S.W. Loke, N.E. Reed, G. Graham eds. : *Advances in Artificial Intelligence*. Lecture Notes in Artificial Intelligence #2112, (Springer ed., 2001), 259-269.
- [6] S. Miles, M. Joy and M. Luck, Designing Agent-Oriented Systems by Analysing Agent Interactions. In P. Ciancarini and M. Wooldridge editors, *Agent-Oriented Software Engineering*, Lecture Notes in Computer Science, (Springer-Verlag 1957, 2001), 171-183.
- [7] G. Wagner, Toward Agent-Oriented Information Systems. Technical report, Institute for Information, University of Leipzig, March 1999. AOIS
- [8] M. Wood and S. A. DeLoach. An Overview of the Multiagent Systems Engineering Methodology. *The First International Workshop on Agent-Oriented Software Engineering (AOSE-2000)*, (Limerick, Ireland, 2000).
- [9] K. Allan, Speech act theory -- an overview. *Concise Encyclopedia of Grammatical Categories* in E. K. Brown & J. Miller eds, (Oxford : Elsevier Science, 1999).
- [10] Kendall, E. A., "Patterns of Agent Analysis and Design," in *Handbook of Agent Technology*, J. Bradshaw, Ed., AAAI Press/ MIT Press, 2000.
- [11] S.K. Card, T.P. Moran and A. Newell, *The Psychology of Human-Computer Interaction*. (Lawrence Erlbaum Ass., Hillsdale, New Jersey, 1983).
- [12] G. De Haan, ETAG-based Design : User Interface Design as Mental Model Specification. In : Palanque and Benyon eds. *Critical Issues in User Interface Systems Engineering*. (Springer Verlag, London, 1996), 81-92.
- [13] S. A. Cerri, Shifting the focus from control to communication : the STReams OBjects Environments model of communicating agents, in J. Padget Eds, *Collaboration between human and artificial societies* vol. 1624, LNAI, (Berlin, Heidelberg, New York : Springer-Verlag, 1999), 71-101.
- [14] S.A. Cerri, V. Loia, and D. Maraschi, Jaskemal : a language for Java agents interacting in Scheme, Presented at *Workshop on Interaction Agents*, (L'Acquila, Italy).

3D-CG Avatar Motion Design by means of Interactive Evolutionary Computation

Hitoshi Iba Nao Tokui Hiromi Wakaki
Graduate School of Frontier Sciences,
The University of Tokyo,
Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan
{iba,tokui,wakaki}@miv.t.u-tokyo.ac.jp.

Abstract. The motion of a 3D-CG avatar is recently used in many games and movies. But it is not easy to generate human motion. Also along with the increasing spread of the Internet, the users want to use various expressions on the web. However the users who don't have special techniques cannot create human motion. In the light of foregoing, the system by which the users can create human motion in an available environment is required.

This paper describes a new approach to generating human motion, more easily and semi-automatically by means of Interactive Evolutionary Computation (IEC). In our system the profile of the avatar is based on the Humanoid Animation standard in order to popularize easily.

1 Introduction

Along with the increasing spread of the Internet in recent years, various expressions are used on the web. Among these one frequently sees expressions that have elements of animation. However, one does not often see expressions using "humanoids" with which we are most familiar. This is because it is not easy to generate human motion.

Also, even in movies, games, and so on, realistic animation that is expressed in 3D space is used. Particularly, in the case of games, animation that is not expressed in 3D space is rare these days. However, it is extremely difficult to generate the motion of humanoids in movies or games, the process of which is difficult to that of drawing a 3D picture. This is because intuition and difficult techniques are required for making drawings by hand to represent motion. Also, it is not possible to fully represent motion by means of motion capture.

Considering these requests, we establish our system. The salient features of our system are as follows.

1. The profile of the avatar based on the Humanoid Animation (hereafter called H-Anim).
2. The arrangement of the system based on Interactive Evolutionary Computation (hereafter called IEC)[11],[1],[5].
3. Genetic representation using linear genetic programming[9].
4. How to express the motion of avatar, i.e., rotation of H-Anim (Orientation Interpolator)[12].

We propose a method of expressing motion and a system that would support creativity and enable general users who have no technical knowledge to easily describe motion.

2 Interactive Evolutionary Computation

Regarding the optimization of a system, there are systems whose optimum performance can be expressed numerically as a fitness function, and systems that cannot be expressed numerically.

For example, when a “favorite” outputting from human-machine interactive systems such as systems that create, process and retrieve images and music, the desired output depends upon the subjectivity of the user, and it is generally difficult to assign a fitness function. Many such fields exist not only in the fields of art and aesthetics, but also in the fields of engineering and education.

One method that is used to optimize such a system is interactive evolutionary computation (IEC)[7],[8]. For example, two representative applications were developed by Sims[6] and Unemi[11].

The procedure for using IEC consists of the following steps:

1. Initialization of a gene group consisting of N individuals using random information
2. Generating and displaying the type of expression of each individual from genes
3. Selecting the desired individual from the displayed individuals, based on the subjectivity of the user
4. Creating genes for N individuals using genetic manipulation, and creating a new gene group, taking the genes of the selected individual as the parent genes.
5. Return to (2).

With the exception of the selection part, the above is exactly the same as the procedure for a general GP [2]. When using a GP, it is necessary to provide 1. a genetic coding method, 2. genetic manipulation, and 3. a procedure for computing adaptability. When using IEC, however, it is important to design a user interface for the user to be able to select the desired individual appropriately, instead of 3.

Because the user makes a decision after seeing each visualized individual, there is an inherent limit to the size of the group (population size) constituting one generation and also the number of generations, so it is not possible to solve large-scale complex by relying on the power of the CPU as in the case of GP.

Also, because the population size is extremely small, IEC has the disadvantage that the solution tends to converge early. When using IEC, it is necessary to configure a setup that avoids this problem.

3 Humanoid Animation

It is expected that the necessity for expressing humanoids in a virtual on-line environment will increase along with the growth of 3D Internet. To this end, the method of expressing a standard humanoid in VRML97 has been specified by the Humanoid Animation Working Group (<http://h-anim.org/>). This is called Humanoid Animation (hereafter called H-Anim). VRML enables humanoids to be displayed on a browser, hence they can be seen on the Internet, and it is anticipated that they will have wide application.

The H-Anim standard determines the name of each part of a humanoid, the method of the joints, and so on. Once the configuration of the body has been determined, it is possible to insert the description of rotation and motion of each part in the same way as for a normal VRML.

However, the parameters that determine the way in which this rotation and motion take place are not values that are readily understood by people. In order to describe this motion, it is absolutely necessary to rely on the power of software that takes in the humanoid described by H-Anim and then writes it as VRML when the parts are moved.

4 Current situation of 3D-CG

At present, 3D-CG animation is frequently used in games and movies. Generally, motion capture is used to take in the motion of a humanoid and impart the same motion to an avatar, or its motion is drawn by hand so as to produce realistic motion.

In 3D-CG description software, for example, a human is divided into various parts and the shape of each part described. Motion is expressed using IK (inverse kinematics)[4] whereby if, for example, an arm is pulled while the relationship between each part of the body is maintained constant, the other joints are moved in such a way as to obtain natural motion.

It is not only difficult to create motion conjured up in the imagination of a person using 3D-CG software, but it is also difficult to conjure up an image of "natural motion". If it is judged as "unnatural motion" sophisticated techniques and a lot of experience are required to know just how to correct it.

Also, looking at the current state of 3D-CG that is used in practice, it can be seen that fine motion is described using manual work. When using a Genetic Programming, it is difficult to search for natural motion in a gravity-free world by using a fitness function alone. Accordingly, if an interactive evaluation is made, resulting in an expression that matches the human sensibility, it is reasonable to say that this is the solution.

For this reason, we have studied the possibility of realizing natural motion by using IEC, for humanoid 3D-CG alone.

5 Proposed system based on IEC

5.1 Genetic Structure

In this research, we determine the method of describing the motion of an avatar used in H-Anim in such a way that the amount of rotation of the joints could be obtained from the genes in GP. Also, we used IEC to enable an evaluation to be made by the user. In this way, we configured a system that could semiautomatically provide better motion, more easily (Figure 1).

The genes use arithmetical operations, triangular functions, min, max, and so on, as nodes, and the variable is time t . By assigning time, the amount of rotation is obtained from the genes of GP for each time that matches the H-Anim description. Of the axes and angles that determine rotation, we made the axes other than those of the hip node and the shoulder node fixed.

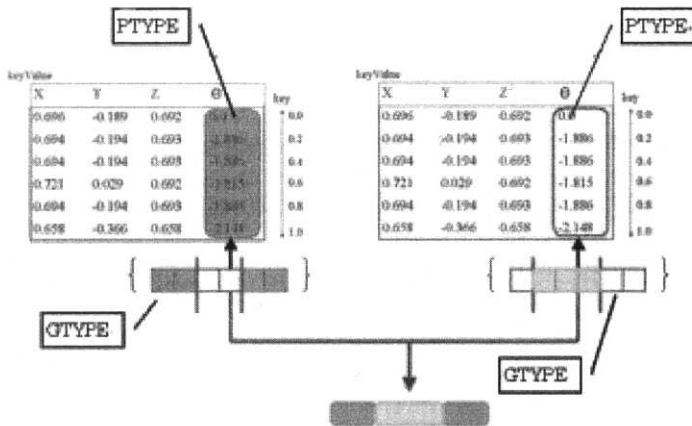


Figure 1: How to obtain the rotation values from the genes in GP.

5.2 Setup of a system that Supports 3D Avatar

When using IEC, users select some individuals to give them high fitness value. But this system enables not only a complete avatar (i.e. individual) but also the position of the avatar to be selected.

When displaying an avatar, we limited the rotation angle of the joints so that the joints would bend in the same way as the joints of the human body. The limit range can be changed to suit the motion that the user wishes to create. For the limit setup, we used a root function to ensure that the motion near the boundary of the limit range (Figure 2) is smooth.

When the number of avatars was simply selected from the initial generation, the motion of each avatar was often similar to the other. Accordingly, we added the work of “initializing unselected avatars”. As a result, the system was improved, enabling as many avatars as possible to be selected up before the generation changed over.

5.3 Outputs from our system

As a output from our system, for example, we obtained the motion indicated in the Figure 3.

It was also possible to make a selection for each part of the avatar, so in the case where “the motion of the entire avatar is not bad, but one part (for example the arm) alone exhibited an unnatural motion”, the element for the part did not enter the selection of individuals as parents. As a result, the range of selection widened, and the intention of the selection mode by the user was more easily refrected in the system.

We added the work of initializing unselected avatars, and modified the program so that initialized avatars were inserted instead of unselected avatars. As a result, it became possible to select many avatars up to the generation changeover. As a result of this work, it was possible to avoid the appearance of motion that rapidly became convergent over several generations. Particularly, we think that because we were able add new avatars from an intermediate gen-

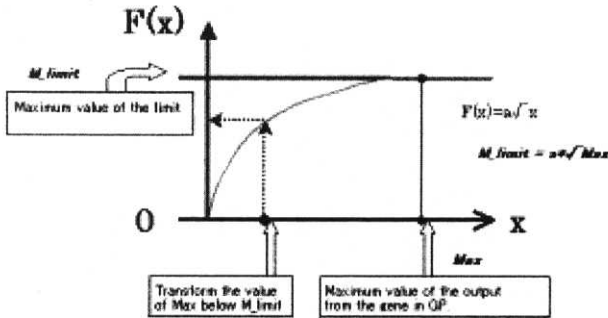


Figure 2: Setup of the function of the rotation limit.

eration, not only the close crossbred of a mere 9 avatars, but also other elements entered the choices reducing to get into a local optimum.

When the users have an image of the motion in some degree, the motion can be kept within the limit range from the outset. So even though the search ranges becomes narrow, it becomes easier to find a solution.

Of the axes and angle that determine the rotation, we fixed the x , y and z of the parts of the avatar other than the hip node and shoulder node. If the specifications are determined so that the bendable axes change through various directions, and the rotational angle also changes widely, the range of human-like motion will be exceeded, and the result will tend to appear to be like that of a puppet. Although the joints in the human body do not move in one direction alone with respect to a single axis, as a simple model we fixed the axis, and also fixed the rotation angle to the GP function, which resulted in a more human-like motion than before.

As a result of the above experiment, we confirmed that we could obtain the motion matched to such topics as “dancing in time to the given music” and “bowing” from a vague image. Particularly, when evolving with a certain type of music, our system shows all individuals with the music. And then, it is enough for user to only select some individuals, the motion of that he or she feels in time to the back music. So even if the user doesn’t have any particular image, the system can evolve better motion than earlier generations. By showing avatars motion, a particular motion may come into his or her mind.

Showing the results of outputs, people might have thought this motion was made as if using digital data of the music. These dances are, for example, “stepping fast”, “lifting a leg slowly”, “waving his body in time to given music”, and so on. But that was all made just as the user feels it in time to the music. So, of course, the user can adopt slow motion from fast music.

An operation demonstration created by this system can be seen freely on the Internet.¹

¹<http://www.miv.t.u-tokyo.ac.jp/~wakaki/avatar/action.html>

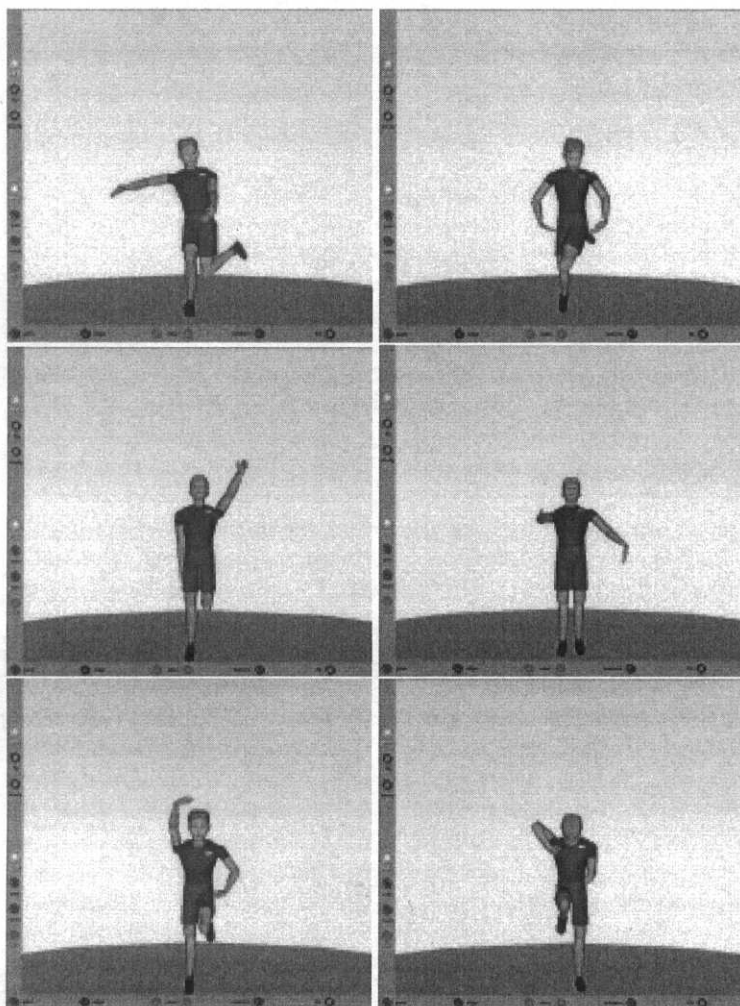


Figure 3: Results of "dancing" from output by this system.

6 Discussion

6.1 Summary of Results

The proposed approach effectively enables the user to construct a sequence of short motions as a dynamic animation. More precisely, we have confirmed the following points empirically.

1. When making a motion in time to a music, the user can generate a motion sequence more easily than by the traditional method or by hand. This is because he or she can evaluate and evolve generated motion while listening to the music.
2. The description is evolvable dependent upon whether or not it is suitable to the subject, even though the user does not have a concrete image of motions. Therefore, the method enhances the user's creativity to some extent.

6.2 Study of the Experiment

The following are necessary when searching for a solution that uses the features of IEC.

1. A genetic structure that enables the intention of the user to be readily incorporated
2. A setup that enables the system itself to readily incorporate the intention of the user
3. The number of individuals is extremely small, so the system is liable to produce a local optimum. To prevent this, the system is to be modified so that the convergence speed is not too high.
4. Parameters are to be changed and limits applied according to the intention of the user, in order to effectively reduce the solution search range.

By adopting functions corresponding to each above point in this system, which uses GP, we were able to obtain better results than before.

1. Smooth motion was obtained by changing from GA to GP. In case of GA, a random value is taken as the initial value, so the resulting motion fluctuates markedly. In the case of GP, on the other hand, when the value is computed from the function, a quantized value is obtained from a continuous function, hence the value does not always undergo a large change. Also, the way in which the value changes depends upon the function, so it can be said to match the sensibility of the user who observes this change as a change in "motion".
2. At the user interface, we modified the system to enable not only avatars but also their parts of the body to be selected. There are 29 nodes, and when the user observes the motion of an avatar, he or she sees the good and bad aspects from the individual parts rather than from each joint, so it is better to select individual parts rather than individual nodes.
3. If the convergence speed is too high, and the motion of all avatars becomes similar to each other after two or three generations, the value of performing an evaluation by user selection will be lost. The avoidance of this situation involves selecting only the avatars that were displayed once, hence is difficult. In which only avatars not selected by the user are initialized and replaced with new individuals. If the user wishes to select only two or

three avatars in the initial generation, the remaining six or seven avatar are initialized. If the number of avatars that were not selected is small, the possibility of individuals that meet the user's expectations being found is reduced. Conversely, if the number is large, it is possible to obtain several individuals that meet the user's expectations by carrying out one or two trials. When seven or more individuals were selected, the number of types of intersections that occurred among them was large, and diversity occurred in the next generation. We think that by carrying out the same trial (initializing unselected individuals), even in subsequent generations, it is possible to include in the group not only individuals created from the same parent but also individuals created from a completely new parent, thus increasing effectiveness.

4. There are two limits that the user can set, a limit on node selection and a limit on the rotation angle of each node. We found that by applying these limits, it was possible to apply irrevocable conditions as the expected motion, and that in the initial generation it was also possible to approach the expected motion than by using motion generated entirely at random.

6.3 Related Research

In the research carried out by Berlanga et al. [3], GA was used to generate better avatar motion. In this research, four avatars were used, and their gestures were selected interactively, enabling their motion to be expressed in a more complex and refined way.

From the outset, each motion possessed by the system was regarded as a gene, and these genes were re-arranged as GA genes, creating a series of motion. This system is created by 10 different simple motions (equivalent to each of the GA genes). By considering the average of 5 genes as a head gene, it is conceivable to create 100,000 (10^5) combinations of different expressions. They insist on the effect that a very large number of expression can be created.

The following methods can be used to change over the generation in a GA.

1. When not even one individual is selected, all individuals are generated randomly.
2. When only one individual is selected, one is copied, and the others are mutated based on the selected one.
3. When two or more individuals are selected, the individual first selected is simply mutated then left alone, and the generation is created by causing the other individuals to intersect each other.

However, it is considered that those methods are limited in their ability to create more complex, human-like expressions. This is because what can be called the components of motion are already determined by the "simple motion of the avatar".

As another IEC application, we have been developing an interactive system called "CONGA" (the abbreviation of "composition in genetic approach" and also the name of an African percussion), which enables users to evolve rhythmic patterns with an evolutionary computation technique [10]. Our system maintains both GA and GP populations and represents music with the combination of individuals in both populations (Figure 4). GA individuals represent short pieces of rhythmic patterns, while GP individuals express how these patterns are arranged in terms of their functions. In this way, we try to express a musical structure, such as a repetition, with the structural expression of GP and evolve longer and more complicated rhythms

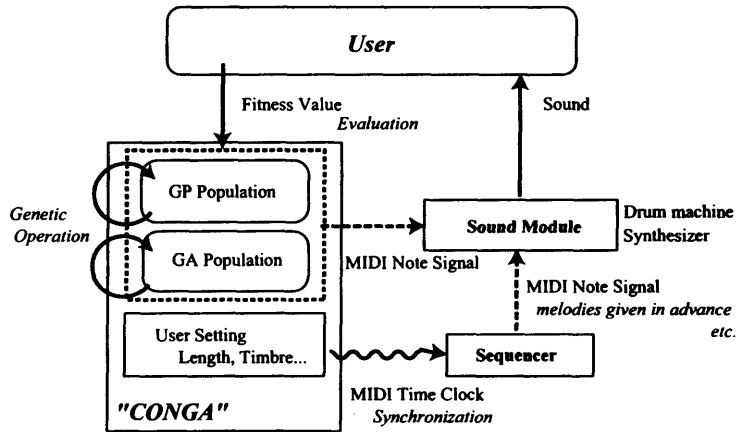


Figure 4: The system architecture.



Figure 5: A typical generated rhythm phrase generated by "CONGA".

without spreading the search space. We have conducted several evaluation experiments so far. Figure 5 shows a typical "rock'n'roll rhythm" generated in this experiment. A couple of generated rhythms in these experiments are available from our web site as sound files (URL: <http://www.miv.t.u-tokyo.ac.jp/tokui/research/iec-music/>).

7 Conclusion and Future Works

The system that we made in this research is intended for motion design using words tht express emotion and words that transmit vague images. Motion design based on words that indicate motion (e.g. "run" and "throw") is left up to tools that can make fine descriptions. At the present time, we think this is an important approach both as a tool that assists the user's creativity, and also as a system proposal that helps the user to create expressions without using difficult technology.

The following are three important topics for the future.

- 1. Efficiently searching for a limitless solution space by applying physical laws such as

gravity and center of gravity balance using some method or other

2. Because the motion of each joint is unrelated to other joints, it is necessary to incorporate the mutual relationship between joints in a genetic expression from the viewpoint of obtaining natural motion.
3. This system will be imported into existing animation tools.

We are going to collaborate with a designer of 3D-CG animation, which will demonstrate the usefulness of our system.

The motion of an avatar appears natural in a gravity-free space. This is because a person normally views the world that he sees with his own eyes from inside that world. It is considered that the solution lies in the ability of a person to select and create motion that matches his sensibility.

At present, with the Internet having spread over an extremely wide range, we think that if the user can create motion for avatars using VRML that can be displayed on a browser, the number of methods of creating individual expressions will increase, in the same way as for pictures, music, and so on. The use of such tools will become indispensable for enabling many people to readily and semiautomatically create the motion of avatars, for personal HP and other applications.

References

- [1] Bentley,P., "Evolutionary Design by Computers", Morgan Kaufmann Publishers Inc., 1999.
- [2] Banzhaf,W., and Nordin,P., Keller,R., and Francone,F., *Genetic Programming – An Introduction* , Morgan Kaufmann Publishers, Inc., 1998.
- [3] Berlanga,A., Isasi,P., and Segovia,J., "Interactive Evolutionary Computation with Small Population to Generate Gestures in Avatars", in *Proc. the Genetic and Evolutionary Computation Conference 2001*, 2001.
- [4] Nearchou A., "Solving the Inverse Kinematic Problem of Redundant Robots Operating in Complex Environments via a Modified Genetic Algorithm", *Journal of Mechanism and Machine Theory*, vol. 3, num. 2, pp. 110-121, 1998.
- [5] Nishino,H., Takagi,H., and Utsumiya,K., "Implementation and Evaluation of an IEC-Based 3D Modeling System", in *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC2001)* , Tucson, AZ, USA, pp.1047-1052, Oct. 7-10, 2001.
- [6] Sims,K., "Artificial Evolution for Computer Graphics", in *Proc. ACM SIGGRAPH'91 Conference* , Las Vegas, Nevada, pp.319-328, July
- [7] Takagi,H., "Interactive Evolutionary Computation - Cooperation of computational intelligence and human KANSEI", in *Proc. 5th International Conference on Soft Computing and Information/Intelligent Systems* , pp.41-50, Oct, 1998.
- [8] Takagi,H., "Interactive Evolutionary Computation: fusion of the capacities of EC optimization and human evaluation", in *Proc. the IEEE*, Vol. 89, num. 9, pp.1275-1296, 2001.
- [9] Tokui,N., and Iba,H., "Empirical and Statistical Analysis of Genetic Programming with Linear Genome", in *Proc. 1999 IEEE International Conference on Systems, Man and Cybernetics (SMC99)*, IEEE Press, 1999.
- [10] Tokui,N., and Iba,H., "Music Composition with Interactive Evolutionary Computation", in *Proc. of Generative Art 2000 (GA2000)*, pp.215-225, 2000.
- [11] Unemi,T., "SBART2.4:Breeding 2D CG images and movies ,and creating a type of collage.",iin *Proc. The Third International Conference on Knowledge-based Intelligent Information Engineering Systems*, pp.288-291, Adelaide, Australia, Aug,1999.
- [12] <http://h-anim.org/>

Alliance Formation with Several Coordinators

Vladimir MARIK, Viktor MASHKOV

Czech Technical University, Technická 2, 166 27 Prague, Czech Rep.
 marik@labe.felk.cvut.cz

Abstract. Practical issue of alliance formation is considered as a subproblem of a more general coalition formation problem in the context of multi-agent systems. The paper describes the process of creating alliances with specific restrictions applied. The intended alliance is formed by several independent coordinators which exchange information during the alliance formation process. The main goal of the research consists in highlighting the problems occurring when alliance is formed by several coordinators and sketching the ways of their solution. The approach proposed integrates the current results in both the fields of the multi-agent research and the complex discrete system diagnostics.

1 Introduction

Coalition formation research is commonly associated with war avoidance operations such as peace-keeping, noncombat evacuations, or disaster relief operations. Unlike classical war operations, where decision-making technology is strictly hierarchical, operations other than war (OOTW) are typically cooperative efforts involving several vaguely organized groups of people (often volunteers). OOTW operations also differ from more hierarchical planning in that they are relatively flexible and dynamic in how they group individual organizations. New entities can freely join an operation autonomously and get involved with planning according to their capabilities. Given this, any organization framework must be essentially "open" [1]. Several experimental frameworks of this type have been already developed, like e.g. system CoAX [2] and CPlanT [3].

Coalition formation seems to be quite a new, emerging subfield of multi-agent systems research. It seems to be quite natural to represent and simulate the coalition formation processes by multi-agent systems in which each complex, organized entity (institution) is represented by an agent. To reduce the complexity of the coalition formation process, it was proposed in [3] to base this process on creating *alliances*, i.e. sets of agents that agree to share some private information and eventually cooperate. The agents forming an alliance are usually represented by a *lead (coordinator)* and group themselves into various, temporary, smaller coalitions to accomplish certain missions.

The paper addresses the problem of formation of a specific category of alliances and is motivated by the development needs of the CPlanT system [3]. An alliance can be defined as a set of agents each of which is familiar with all the remaining ones, trusts them and possesses the necessary information Ω_A for carrying out joint activities aimed at achieving a certain goal or goals.

It means that the introduced alliance should meet the following two conditions:

- 1) every agent A in the alliance Θ should know all the other agents of the alliance (i.e. structure of the alliance). This condition can be expressed as

$$\forall A, A \in \Theta : \alpha(A) = \Theta,$$

where $\alpha(A)$ – a set of all agents that agent A is aware of;

- 2) every agent A in the alliance Θ should be confident that every other agent of the alliance possesses information about the alliance structure. This condition can be expressed as

$$\forall A, A \in \Theta : \beta_k(A) = \Theta,$$

where $\beta_k(A)$ – a set of all agents that share their knowledge about the alliance structure with the agent A .

Formation of such alliance can be performed in different ways, one of which is based on centralized approach [4]. In this case, the sought alliance is formed by just one central coordinator. In some cases, this approach shows certain drawbacks, namely low efficiency, and namely sensitivity to the reliability of the central coordinator.

There is suggested another approach to forming the sought alliance here, which is based on distributed communication processes. This approach is based on using several subsidiary coordinators for the alliance formation. The benefits of such an approach to alliance formation are as follows:

- Distributed information delivery presumes that communication among the agents can be performed simultaneously. Besides, the seeking for an alliance candidate is fulfilled not only by the central coordinator, but also by any agent which is already involved in the process of the alliance formation.
- Using of several coordinators allows to reduce the dependence of alliance formation process on reliability of every single coordinator.
- In view of the fact that several coordinators can simultaneously communicate with several potential alliance candidates, the number of agents refusing to join the alliance has a lesser impact on the process of alliance formation.

In its turn, the idea of the alliance formation process based on several coordinators invokes many additional problems. In this paper, we concentrate mainly at the problem connected with diagnosing information transfers (messages) which have been performed incorrectly during the alliance formation process (i.e. *diagnosing information alterations*).

We assume that any communication failure leads to an information alteration [5]. Under the *information alteration* we understand the event when the recipient agent receives information differing from the originally intended information of the sender agent.

The problem studied in this paper is the key point for protecting the alliance formation processes from misunderstandings, corruptions of any kind as well as from potential intrusion into the communication network.

2 General description of the alliance formation process with several coordinators

Any agent can initiate the alliance formation process. The agent which starts the process of alliance formation is considered as a central coordinator (CC). We assume that any agent which accepted the proposal to become a candidate for the alliance after communication with CC, in its turn, can perform the role of a coordinator. We consider such agent as a subsidiary coordinator (SC). An agent becomes SC as soon as it communicates with any other agent (with the exception of CC or SC). Having become SC, the agent has two options: either to proceed seeking agents for alliance or to send information about agents to all coordinators (CC and SCs) which are known to it.

With the account of the fact that the process of alliance formation is random and the number of potential candidates for alliance can be arbitrary at any moment, we suggest that the newly created SC should periodically inform other coordinators about the candidates that it succeeded to find. This statement is justified by the fact that seeking a new agent for the alliance may be a redundant process. Moreover, every newly created SC can be a source of incorrect information. Periodical exchange of information among coordinators enables

each of them to check whether the alliance is being formed in a correct way. During the communication among the selected SC and all other coordinators which are known to it, the selected SC receives information about all the potential candidates for the alliance. Then, the selected SC informs all the agents which it communicates with about all other potential candidates for the alliance. On the basis of this information, every such agent can make decision about its further participation in the alliance (see Fig. 1). Any agent can leave the alliance if it doesn't want (from any reason) to cooperate with an agent-newcomer.

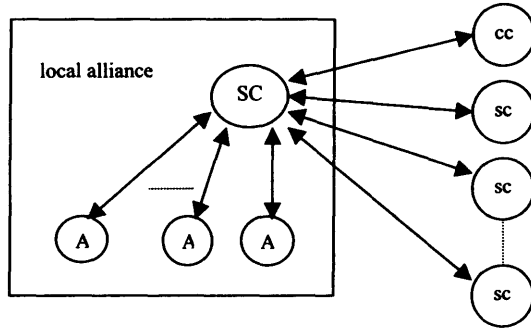


Fig. 1. Alliance and local alliance

In the situations when either one of the communicated SC_s or one of the agents communicated by the selected SC refuses to join the alliance, the given SC ignores them. The SC, after communicating with all the coordinators which are known to it, proceeds with seeking a new candidate for the alliance. As soon as the SC finds a new candidate, it forms a new local alliance (see Fig. 1) which includes all the agents which it communicated with. The optimal procedure for forming such a local alliance is presented in the paper [4]. After that, the SC sends updated information to all other coordinators.

It may happen that one of the coordinators distributes incorrect information. Such situation may occur due to a variety of reasons (e.g. design faults; interaction faults; internal, intentionally malicious faults; intrusions etc.). In order to provide detections of information alterations and identification of the coordinator which is the source of incorrect information, it is necessary for the coordinators to exchange diagnostic information among themselves.

For every coordinator, the diagnostic information received from the other coordinators is the initial data for a diagnostic (detection) algorithm. We assume that every coordinator - after receiving a new piece of information - performs the diagnostic algorithm and plans its further actions according to the obtained result. It is important to ensure that the coordinators which detected an untrustworthy coordinator, would act jointly to isolate the untrustworthy coordinator. Provided the untrustworthy coordinator is detected by majority of coordinators, the sought alliance can be formed correctly.

It is desirable that every reason leading to a communication failure should be identified. However, this is a complex task. In our research, we make an attempt to identify at least some of these reasons. To solve this task we suggest the approach based on *information comparisons* (i.e. on comparing of the messages' contents) performed by individual agents.

3 Diagnosing performed by a coordinator

During the alliance formation process, the coordinators exchange information which comprises the following two parts:

1. Information related to the alliance to the alliance formation process;

2. Information being used for diagnostic purposes.

The former part includes:

- Information about the intended alliance (Ω_{FA}) (e.g. the alliance goal, the total number of agents required for the alliance, the expected duration of its operation, etc.)
- Information about all the agents which, by this time, have agreed to be candidates for the alliance (Ω_{cand}).

Every coordinator compares the information Ω_{FA} which it sends or receives to/from other coordinators and produces a corresponding outcome on the basis of this comparison.

The latter part of the information exchange between the coordinators includes:

- information about outcomes R_i , $i \in 1, n$ of comparing the information Ω_{FA} "owned" by different coordinators. There are several kinds of such comparisons (in the following text the information received is noted as ${}_R\Omega$, the information sent as ${}_S\Omega$ and the information which the agent is going to send as Ω):

- 1) Two items of information received by the coordinator № i from any two other coordinators ${}_R\Omega_{i,FA}(j)$ and ${}_R\Omega_{i,FA}(k)$ (e.g. from the coordinators № j and № k):
 $R_i^1 = 0$, if ${}_R\Omega_{i,FA}(j)$ and ${}_R\Omega_{i,FA}(k)$ are identical; otherwise $R_i^1 = 1$. (1)
- 2) The item of information which the coordinator № i has sent (${}_S\Omega_{i,FA}$) with the item of information it received later (${}_R\Omega_{i,FA}$):
 $R_i^2 = 0$, if ${}_S\Omega_{i,FA}$ and ${}_R\Omega_{i,FA}$ are identical; otherwise $R_i^2 = 1$. (2)
- 3) The item of information which the coordinator № i received (${}_R\Omega_{i,FA}$) with the item of information which it sent (${}_S\Omega_{i,FA}$):
 $R_i^3 = 0$, if ${}_S\Omega_{i,FA}$ and ${}_R\Omega_{i,FA}$ are identical; otherwise $R_i^3 = 1$. (3)
- 4) The two items of information received by the coordinator № i from the same coordinator № j in the time slots t and $t+1$:
 $R_i^4 = 0$, if ${}_R\Omega_{i,FA}(j_t)$ and ${}_R\Omega_{i,FA}(j_{t+1})$ are identical; otherwise $R_i^4 = 1$. (4)
- 5) The item of information which the coordinator № i received (${}_R\Omega_{i,FA}$) with the item of information which it is going to send ($\Omega_{i,FA}$):
 $R_i^5 = 0$, if ${}_R\Omega_{i,FA}$ and $\Omega_{i,FA}$ are identical; otherwise $R_i^5 = 1$. (5)
- 6) The item of information which the coordinator № i has sent (${}_S\Omega_{i,FA}$) with the item of information it intends to send ($\Omega_{i,FA}$):
 $R_i^6 = 0$, if ${}_S\Omega_{i,FA}$ and $\Omega_{i,FA}$ are identical; otherwise $R_i^6 = 1$. (6)

While performing information comparisons every coordinator may make a mistake. Such mistake may be a consequence of incorrect preparing of the contents of messages or of incorrect perception of messages by the coordinators.

Every coordinator after receiving all this information from other coordinators processes it. Such processing consists of the following steps:

- 1) Every coordinator (let's suppose the coordinator № i) compares the information Ω_{FA} according to (1+6) and determines the set of outcomes $\{R_i^m\}$, $m \in 1, \dots, 6$.
- 2) The coordinator № i forms the table which establishes the correspondence between considered information alterations $\{I_i\}$ and outcomes $\{R_j^m\}$ which could be obtained (as a result of analysis) by other coordinators, that is

$$T: [\{I_i\} \times \{R_j^m\}]$$

For forming such a table locally, every coordinator should use the information about performed information transfers $\Omega_{i(coord k \rightarrow coord j)}$ during the last session.

Under the session we understand the lapse of time between two sequential coordinator's communications with agents during which it communicates with some of the other coordinators. The session starts at the moment when the coordinator stops communicating with agents and enters into communication with coordinators. The session finishes when the coordinator turns back to communication with agents.

The lines of the table correspond to considered potential information alterations. Initially, we will consider single information alterations. The columns of the table correspond to outcomes of information comparisons performed by coordinators. The formed table lies in the basis of the algorithm of determining information alterations which occurred during the alliance formation process.

In order to introduce the main concepts underlying the diagnostic algorithm, we start with a simple example.

Let's assume that two agents send a piece of information Ω to each other as it is shown in Fig. 2. Subscript at Ω means the ordering number of the information transfer.

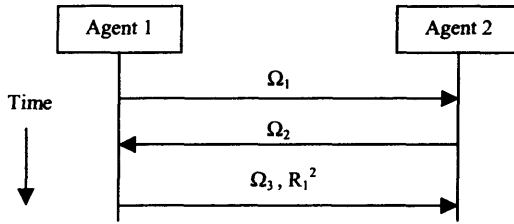


Fig. 2. Communication scenario

The agent 1 can compare the information Ω_1 which it sent to the agent 2 with the information Ω_2 which it received from the agent 2. On the basis of this comparison, the agent 1 forms the outcome R_1^2 according to the expression (2). Then the agent 1 sends to the agent 2 the outcome R_1^2 together with the information Ω_3 . The agent 2, in its turn, forms the outcome R_2^4 according to the expression (4).

For this simple example with two agents, the following three types information alterations: Ω_1 , Ω_2 , and Ω_3 are possible. We denote the information alteration as I_i , $i=1, \dots, 3$. It is assumed that the information alteration I_i can take the following values: $I_i = 0$ if the recipient agent receives correct information, $I_i = 1$ otherwise.

In the case when one of the information alterations takes place during the information exchange between the agents 1 and 2, one or two outcomes (R_1^2 and R_2^4) will not be equal to "0". Every agent can form the table which reflects the correspondence between possible information alterations $\{I_i\}$ and outcomes $\{R_j\}$ which could be obtained in the case of these information alterations. For the example under consideration, the agent 2 forms locally the following table 1.

Table 1

	R_1^2	R_2^4
I_1	1	1
I_2	1	0
I_3	0	1

The set of obtained outcomes $\{R_1^2, R_2^4\}$ is a syndrome in the terminology used in the area of complex discrete systems diagnostics. Table 1 represents the basis of diagnosing information alterations which may occur during the information exchange between the agents 1 and 2.

For the given table let's consider the following definitions:

Definition № 1: The table line is called *unique*, if it can not be obtained as a result of the elementwise Boolean addition of any combination of lines of the given table.

Definition № 2: The i -th table line is called *dominate*, if for any unequal to it j -th line ($j \neq i$) the following condition is met: $a_{ik} \geq a_{jk}$, $k \in [1, \dots, r]$, where a_{ik} and a_{jk} are elements of the table, i and j - numbers of lines and k is the number of columns, r is the number of outcomes.

Definition № 3: The table line is called *distinguishable*, if there is no line in the given table, whose elements are equivalent to the elements of the given line.

The concepts of uniqueness, dominance and distinguishability of table lines are used for diagnosing the information alterations I_i .

We also will use the following three concepts for diagnosing information alterations I_i : *conditional*, *definite* and *final diagnosing*.

Under the *conditional diagnosing* of information alteration I_i we understand such diagnosing, which allows to detect the information alteration provided a certain condition is met. When determining the information alteration during the information exchange between agents, such a condition can be (for example) like this: the total number of information alterations does not exceed t .

The conditional diagnosing resembles t -diagnosability used in models of interaction (in particular, in the model of Preparata [6]). Thus, as a result of the conditional diagnosing all the information alterations will be detected correctly, provided that their total number does not exceed t . The conditional diagnosing is based on the theorem № 1.

Theorem № 1. For *conditional diagnosability* of information alteration I_i , it is necessary and sufficient that the line of the table, corresponding to the given information alteration, is distinguishable.

For the considered example (see Fig. 3), all the single information alterations will be detected correctly, provided the total number of information alterations is no more than one. This follows from the fact that all the lines of table 1 are distinguishable.

Under the *definite diagnosing* of the information alteration I_i we understand such diagnosing when, as a result of diagnosing, the information alteration I_i is detected definitely, although it remains uncertain if this alteration was the only one and no other alteration(s) occurred. The concept of the definite diagnosing is based on the theorem № 2.

Theorem № 2. For *definite diagnosability* of information alteration I_i it is necessary and sufficient that the line of the table, corresponding to the given information alteration, is distinguishable and unique.

For the considered example (see Fig. 3), just two information alterations I_2 and I_3 meet this requirement. This follows from the fact that the lines of the table 1, corresponding to the given information alterations, are distinguishable and unique.

Under the *final diagnosing* of information alteration I_i we understand such diagnosing when not only information alteration I_i is detected correctly, but also all other information alterations which have occurred are detected as well. The final diagnosing is based on the theorem № 3.

Theorem № 3. For the *final diagnosability* of all information alterations it is necessary and sufficient that the lines of the table, corresponding to the given information alterations, are distinguishable, unique and non-dominate.

For the considered example (see Fig. 2), two single information alterations I_2 and I_3 are finally diagnosable. It follows from the fact that the lines of the table 1, corresponding to the given information alterations, are distinguishable, unique and non-dominate. Whereas the information alteration I_1 is only conditionally diagnosable in view of the fact that the line of the table 1, corresponding to this information alteration, is only distinguishable. Thus, the table 1 enables the agent 2 to determine which single information alterations are conditionally, definitely and finally diagnosable.

The proofs of the theorems № 1, 2 and 3 are presented in [7].

Now, let us proceed with considering the process of alliance formation with several coordinators when some of information transfers are altered. Let's assume that during the alliance formation process there were created only two subsidiary coordinators (SC_1 and SC_2) (see Fig. 3). Let's also assume that during the last session there were accomplished the following six information transfers between coordinators: ($CC \rightarrow SC_1$); ($SC_1 \rightarrow CC$); ($SC_1 \rightarrow SC_2$); ($SC_2 \rightarrow SC_1$); ($CC \rightarrow SC_2$); ($SC_2 \rightarrow CC$) and these information transfers were performed in the sequence presented (see Fig. 4). For these information transfers we introduce the following denotations of their possible alterations:

$I_1 = (CC \sim \rightarrow \sim SC_1)$; $I_2 = (SC_1 \sim \rightarrow \sim CC)$; $I_3 = (SC_1 \sim \rightarrow \sim SC_2)$;

$I_4 = (SC_2 \sim \rightarrow \sim SC_1)$; $I_5 = (CC \sim \rightarrow \sim SC_2)$; $I_6 = (SC_2 \sim \rightarrow \sim CC)$;

During the information exchange with other coordinators every coordinator received a set of outcomes $\{R_i^m\}$, $m \in 1, \dots, 6$. The considered outcomes have received the following denotations: R_1 represents the outcome formed by CC; R_2 by SC_1 ; R_3 by SC_2 ;

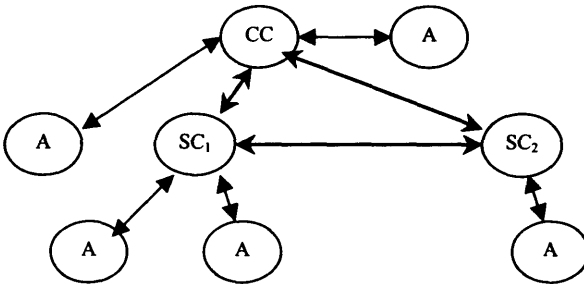


Fig. 3. An example of an alliance with three coordinators

SC replies to the request from another SC by sending back the information Ω_{FA} which was distributed earlier (i.e. it had received earlier). So, for example, SC_2 having received information from SC_1 , replies with information received earlier from CC . Taking into account these denotations, every coordinator forms a table which reflects correspondence between possible information alterations I_i , $i \in 1, \dots, 6$ and outcomes R_j , $j \in 1, \dots, 3$.

We assume that the coordinator firstly forms the simplified table which includes only part of the considered outcomes. Namely, the outcomes R_i^1 and R_i^2 , $i \in 1, \dots, 3$.

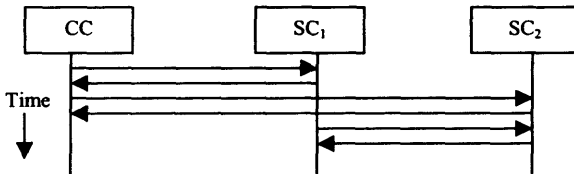


Fig. 4. Expected sequence of communication transfers

Therefore, for the example, under this consideration SC_1 forms the following table 2. On the basis of this table, SC_1 can infer that the information alterations I_1 , I_2 and I_5 are finally diagnosable, I_3 and I_4 are definitely diagnosable, whereas information alteration I_6 is not diagnosable at all.

As a result, the following expression can be obtained from the table 2:

$$(I_2 R_1^2 \vee (I_1 \vee I_3 \vee I_4 \vee I_5 \vee I_6) \bar{R}_1^2) \cap ((I_1 \vee I_4) R_2^1 \vee (I_2 \vee I_3 \vee I_5 \vee I_6) \bar{R}_2^1) \cap ((I_3 \vee I_5) R_3^1 \vee (I_1 \vee I_2 \vee I_4 \vee I_6) \bar{R}_3^1) \cap ((I_3 R_3^2 \vee (I_1 \vee I_2 \vee I_4 \vee I_5 \vee I_6) \bar{R}_3^2) \cap (I_4 R_2^2 \vee (I_1 \vee I_2 \vee I_3 \vee I_5 \vee I_6) \bar{R}_2^2) \quad (7)$$

Table 2

	R_1^2	R_2^1	R_2^2	R_3^1	R_3^2
I_1	0	1	0	0	0
I_2	1	0	0	0	0
I_3	0	0	0	1	1
I_4	0	1	1	0	0
I_5	0	0	0	1	0
I_6	0	0	0	0	0

The given expression is obtained from the following analysis: In accordance with the table 2, the outcome R_1^2 will be equal "1" in the case of availability of the information alteration I_2 and is equal to "0" if there are the following information alterations: I_1, I_3, I_4, I_5 and I_6 . Thus, the expression

$$I_2 R_1^2 \vee (I_1 \vee I_3 \vee I_4 \vee I_5 \vee I_6) \bar{R}_1^2 \quad (8)$$

will determine those single information alterations which do not contradict the received outcome R_1^2 .

Similarly, for the outcomes R_2^1, R_3^1, R_3^2 and R_2^2 the corresponding expressions could be obtained:

$$(I_1 \vee I_4) R_2^1 \vee (I_2 \vee I_3 \vee I_5 \vee I_6) \bar{R}_2^1 \quad (9)$$

$$(I_3 \vee I_5) R_3^1 \vee (I_1 \vee I_2 \vee I_4 \vee I_6) \bar{R}_3^1 \quad (10)$$

$$I_3 R_3^2 \vee (I_1 \vee I_2 \vee I_4 \vee I_5 \vee I_6) \bar{R}_3^2 \quad (11)$$

$$I_4 R_2^2 \vee (I_1 \vee I_2 \vee I_3 \vee I_5 \vee I_6) \bar{R}_2^2 \quad (12)$$

As a result of intersection of the expressions (8÷12), we obtain the main expression (7) for the diagnosing of single information alterations.

SC_1 receives actual (real) outcomes from CC and SC_2 (i.e. R_1^2, R_3^1 and R_3^2) and using the expression (7) determines the information alteration which occurred during the alliance formation. When the obtained real syndrome doesn't match any syndrome presented in the table 2, the SC_1 can proceed with considering faulty situations with multiple information alterations.

Firstly, SC_1 considers situations with two different information alterations. For this purpose, SC_1 forms the table 3. The table 3 contains the lines corresponding to all possible double information alterations. This table doesn't contain those double information alterations which comprise the information alteration I_6 . It follows from the fact that if the information alteration I_6 is not diagnosable, then all its combinations with any other considered information alterations will be also not diagnosable. The table 3 is derived from the table 2 by pairwise elementwise Boolean addition of its lines.

Analysis of the table 3 allows SC_1 to come to the conclusion that situations with the double information alterations $I_1 I_2, I_1 I_5$ and $I_2 I_5$ are finally diagnosable. This follows from the fact that the lines of the table 3 corresponding to the given situations are

distinguishable, unique and non-dominate. The table 3 also allows to derive expression (13) which should be used for diagnosing double information alterations.

$$((I_1I_2 \vee I_2I_3 \vee I_2I_4 \vee I_2I_5) R_1^2 \vee (I_1I_3 \vee I_1I_4 \vee I_1I_5 \vee I_3I_4 \vee I_3I_5 \vee I_4I_5) \bar{R}_1^2) \cap ((I_1I_2 \vee I_1I_3 \vee I_1I_4 \vee I_1I_5 \vee I_2I_4 \vee I_3I_4 \vee I_4I_5) R_2^1 \vee (I_2I_3 \vee I_2I_5 \vee I_3I_5) \bar{R}_2^1) \cap ((I_1I_4 \vee I_2I_4 \vee I_3I_4 \vee I_4I_5) R_2^2 \vee (I_1I_2 \vee I_1I_3 \vee I_1I_5 \vee I_2I_3 \vee I_2I_5 \vee I_3I_5) \bar{R}_2^2) \cap ((I_1I_3 \vee I_1I_5 \vee I_2I_3 \vee I_2I_5 \vee I_3I_4 \vee I_3I_5 \vee I_4I_5) R_3^1 \vee (I_1I_2 \vee I_1I_4 \vee I_2I_4) \bar{R}_3^1) \cap ((I_1I_3 \vee I_2I_3 \vee I_3I_4 \vee I_3I_5) R_2^3 \vee (I_1I_2 \vee I_1I_4 \vee I_1I_5 \vee I_2I_4 \vee I_2I_5 \vee I_4I_5) \bar{R}_3^2).$$

In order to identify the double information alterations (i.e. situations with two different independent alterations) which occurred during the latest session, SC₁ substitutes the real syndrome in expression (13). If the obtained real syndrome doesn't match any syndrome presented in the table 3, the SC₁ can proceed with considering triple information alterations.

Table 3

	R_1^2	R_2^1	R_2^2	R_3^1	R_3^2
I_1I_2	1	1	0	0	0
I_1I_3	0	1	0	1	1
I_1I_4	0	1	1	0	0
I_1I_5	0	1	0	1	0
I_2I_3	1	0	0	1	1
I_2I_4	1	1	1	0	0
I_2I_5	1	0	0	1	0
I_3I_4	0	1	1	1	1
I_3I_5	0	0	0	1	1
I_4I_5	0	1	1	1	0

The procedure of investigating the triple information alterations is the same as the above considered procedure for investigating the double information alterations. Multiple information alterations need to be considered when matching between the real syndrome and the syndromes presented in the table 2 hasn't been found by SC₁.

However, there is a possibility that SC₁ has found such matching but is unable to identify the reason of the faulty situation (e.g. in the case of the information alterations I_3 , I_4 or I_6). In the given case, SC₁ can take into account other outcomes obtained (calculated) by other coordinators (e.g. the outcome R_3^3). The coordinator SC₁ can update the table 2 and form a new table with additional columns. The updated table 2 in this case includes one additional column R_3^3 which has zero in every line except one "1" in the 5-th line (i.e. in the line I_5). Now SC₁, using the updated table 2, can finally diagnose not only the information alterations I_1 , I_2 , I_5 but also I_3 .

Every coordinator can also take into account the results of diagnosis produced by other coordinators and sent to it during the session. In our example, during the session SC₂ forms the table 4 in the same way as it was done by SC₁.

The table 4 and the real syndrome (i.e. the set of outcomes received by SC₂) represent the basis of the diagnosis performed by SC₂. The coordinator SC₂ sends the result of the performed diagnosis RS (SC₂) to all other coordinators known to it. In our example, it sends the result of diagnosis to coordinator SC₁. The coordinator SC₁, in its turn, having received the result RS(SC₂), can compare it with its own diagnosis result RS(SC₁). Generally, the results received from other coordinators should help the coordinator to identify faulty situations. Although, should there be any contradiction in the results, the coordinator ought to take decision as to what extent it can believe and accept the results (or even outcomes) received from other coordinators.

This problem presents a separate complex task which is the subject of our current research.

There may occur situations when one or several coordinators obtain the diagnostic result testifying to multiple information alterations. Moreover, all the altered information transfers are related to one and the same coordinator. These situations can be identified as faulty situations with a coordinator distributing incorrect information. Thus, the result of diagnosis obtained by the coordinator could point either at concrete information transfers which were altered or at concrete coordinators which distribute incorrect information.

Table 4

	R_1^2	R_3^1	R_3^2	R_3^3
I_1	0	0	0	0
I_2	1	0	0	0
I_3	0	1	1	0
I_5	0	1	0	1
I_6	0	0	0	0

In the former case, the coordinator should inform all the coordinators known to it about detected information alterations. It will be sufficient that all the coordinators take this information into account in their analysis and proceed with distribution of correct information.

In the latter case, the coordinator should impart the result of its analysis to reliable coordinators only. For forming alliance in this case, it is important that the reliable coordinators cooperate and ignore the unreliable ones. If the total number of unreliable coordinators is less than the number of reliable ones, then we can expect that the sought alliance will be formed in a correct way.

4 Related work

In contradiction to "classical" coalition-formation approaches [8], [9] which consider centralized approach to forming a coalition, our potential application field (OOTW) requires namely higher flexibility and higher information assurance. These requirements can be achieved just by carrying out the coalition-formation process in a more decentralized way. We have decided to create alliances of agents eager to share their knowledge and to cooperate, and to compose the coalitions to accomplish a certain mission from these already pre-prepared alliances. To make the process of alliance formation even more distributed, we considered participation of several coordinators in the process of the alliance formation.

In our paper, we focused our attention mainly to the problem of detecting failures (information alterations) of any kind during the alliance formation process. We suppose that during the alliance formation there may occur multiple information alterations.

It should be noted that, as distinct from known approaches to diagnosing multiple faults [10], [11], [12], our approach is based on the assumptions that the initial diagnostic data are received by the coordinator spontaneously and the coordinator isn't aware in advance of all other coordinators which may communicate with it. That means, the structure of the global system changes dynamically and it is not known in the full extent to any of the

coordinators. In the “classical” multiple fault diagnostics, however, it is generally assumed that the structure of the system is known and diagnostic data are being obtained according to a predetermined algorithm.

According to de Kleer and Williams [10], after performing measurement and obtaining diagnostic information, a diagnostician can select candidates which should be remained for the next consideration. Then, on the basis of both the structure of the investigated system and the obtained diagnostic information, the diagnostician plans his/her next steps (actions) to determine the single candidate which caused the faulty situation in the system. De Kleer and Williams developed the approach and the corresponding algorithm which allows a diagnostician at every next step to consider the minimal set of candidates. This approach could be used by us to simplify the tables involving situations with the multiple information alterations. Particularly, in the table with single information alterations, we could select those information alterations which have definitely not occurred. This way, all the double information alterations (comprising the single information alterations which have definitely not occurred) can be eliminated from the table involving the double information alterations. However, for the small number of coordinators, this simplification is insignificant.

As distinct from [10], we classify all the considered information alterations (which can be called *candidates*) into three classes: finally, definitely and conditionally diagnosable. In our case, every coordinator only analyses the diagnostic information which it possesses and plans no additional actions for clarifying the situation. That is why, the information alterations should be sorted in the most complete way so that the coordinator would be able to determine with greater precision the information alterations occurred during the alliance formation process. In any case, the de Kleer’s approach will motivate our future research and the relationships between this and our approach should be studied in more detail.

Our approach to diagnosing information alterations has some elements in common with the self-diagnosing of digital systems (system level diagnosis) which starting from the paper by [6] has been developed later [13], [14], [15]. Our approach is distinct from the self-diagnosis of digital systems in that the diagnostic data are obtained as a result of the information comparing, whereas with the self diagnosis approach the diagnostic data are obtained as a result of mutual checks between system elements.

5 Conclusions

Although the approach to form an alliance based on centralized information delivery (i.e. when only one central coordinator is used) needs the minimal number of messages to form an alliance, there are situations when this central coordinator doesn’t manage to cope with the complexity of the task just in time. It is expected that in such complex situations the time needed for the alliance formation process can be reduced by using several subsidiary coordinators, each of which can perform the functions of the central coordinator. This vision might be – from the implementation point of view - supported by the FIPA compliant approach: the agents placed on a single platform can be represented by an (extended) directory facilitator which could play the role of the subsidiary coordinator communicating on behalf the members of the alliance located on the given platform.

It is also important that our approach to the alliance formation problem doesn’t depend considerably on the coordinator’s reliability and on all possible faults and intrusions which may occur during the information transfers.

However, on the other hand, this approach can lead to the situation when one or more coordinators (due to some reasons) will distribute incorrect information and, in view of this, will form the alliance in a wrong way. Hence, it becomes significant to solve the problem how to provide the diagnosis of all such untrustworthy coordinators and all the altered

information transfers before a faulty alliance is created. Such diagnosis should be performed by every coordinator with the subsequent exchange of results.

In this paper we have considered diagnosing of information alterations. Admittedly, it should be considered as the first step in the entire diagnosing procedure towards detecting coordinators which distribute incorrect information. Our future efforts will be devoted to this problem which seems to be very important for the application area of OOTW.

Acknowledgments

The EOARD/U.S.Air Force Research Lab contract No. F61775-00-WE043 and the Ministry of Education, Youth and Sports of the Czech Republic, Grant No. LN00B096 supported our research.

References

- [1] Tate, A., Bradshaw, J.M., Pechoucek, M.: Knowledge Systems for Coalition Operations. *IEEE Intelligent Systems*, 7(3), May/June, 2002, 14-16.
- [2] Allsopp, D., N., at al.: Coalition Agents Environment: Multiagent Cooperation in International Coalitions. *IEEE Intelligent Systems*, 7(3), May/June, 2002. 26-35.
- [3] Pechoucek, M., Marik, V., Barta, J.: A Knowledge-based Approach to Coalition Formation. *IEEE Intelligent Systems*, 7(3), May/June, 2002, 17-25.
- [4] Mashkov, V., Marik, V.: Alliance Formation Process and Communication Traffic. *Proc. of IASTED ALA'2002 Conference*, Malaga, 2002.
- [5] Dobson, J.: Inter-agent Communication Model. In: Reference Model and Use Cases (MAFTIA Project IST-1999-11583), Chapter 4, 98-103, 2000.
- [6] Preparata F.,P., Metze G., Chien R.,T.: On the Connection Assignment of Diagnosable Systems. *IEEE Trans. EC.*, vol. EC-16 (1967), 848-854.
- [7] Mashkov, V.: Intrusion Detection in Multi-agent Systems. Technical Report , Gerstner Lab, Czech Technical University, Prague, 2001, ISSN 1213-3000.
- [8] Tambe, M.: Towards Flexible Teamwork. *Journal of AI Research*, 7(1997), 83-124.
- [9] Kaminka, G.A., Pynadath, D.V., Tambe M.: Monitoring Deployed Agent Teams. *Proc. of the 5th Int. Conf. on Autonomous Agents (Agents 2001)*, Montreal, Canada, 2001.
- [10] de Kleer, J., Williams, B. : Diagnosing Multiple Faults. *Artificial Intelligence*, 32 (1987), 97-130.
- [11] Provan, G.: A Model-based Diagnosis Framework for Distributed Embedded Systems. Workshop *Advances in Diagnostics*, Toulouse, 2002.
- [12] Frohlich, P., de Mora,I., Nejd, W., Schroder, M.: Diagnostic Agents for Distributed Systems. In: *ModelAge Workshop*, 1997, 173-186.
- [13] Barsi, T., Grandoni, T., Maestrini, P.: A Theory of Diagnosability of Digital Systems. *IEEE Trans. on Comp.*, Vol. C-25 (1976), 585-593.
- [14] Vedeshenkov, V.: On Organization of Self-diagnosable Digital Systems. *Automation and Computer Engineering*, 7 (1983), 133-137.
- [15] Mallela, S., Masson, G.: Diagnosable Systems for Intermittent Faults. *IEEE Trans. on Comp.*, Vol. C-27, (1978), 379-384.

This page intentionally left blank

Section 10

Web Computing

This page intentionally left blank

Balancing Volume, Quality and Freshness in Web Crawling

Ricardo Baeza-Yates Carlos Castillo

Center for Web Research
Dept. of Computer Science, University of Chile
Blanco Encalada 2120, Santiago, Chile
E-mail: {rbaeza, ccastill}@dcc.uchile.cl

Abstract. We describe a crawling software designed for high-performance, large-scale information discovery and gathering on the Web. This crawler allows the administrator to seek for a balance between the volume of a Web collection and its freshness; and also provides flexibility for defining a quality metric to prioritize certain pages.

1 Introduction

Search engines are the most used tools to find Websites and information in the Web, and they account for a significant percentage of through traffic in many Web sites. These engines have three standard components: crawlers, spiders or robots (input), collection and index (storage) and query resolver and interface (output). For more information we refer the reader to [7, 6].

Nowadays is difficult to build a crawler that can accurately traverse all of the Web because of its volume, pace of change and growth. In fact, Web crawlers have to deal with a lot of challenges at the same time:

- a) They must keep fresh copies of Web pages, so they have to revisit some of them, but at the same time they must discover new pages.
- b) They must use the available resources such as network bandwidth to the maximum extent, but without overloading Web servers as they visit them.
- c) They must get a lot of “good pages”, but they cannot exactly know in advance which ones are the good ones.

In [15] we presented a model that tightly integrates crawling with the rest of a search engine and gives a possible answer about how to deal with these (contradicting) goals. This model generalizes many particular cases, and leads to a new crawling software architecture that has been evolving and that we describe in this paper by focusing in its design issues.

The rest of the paper is organized as follows. The next section covers previous work. Then we describe the software architecture of our crawler followed by the description of the main scheduler, how to be polite to Web servers and some performance issues. We conclude with future planned work.

2 Previous work

Crawlers are an important part of some search engines, and as such, their internals are business secrets, and when they publish information, there is an important lack of details that prevents other to reproduce the work.

Some descriptions of crawlers (in chronological order) are: RBSE [26], Internet Archive [14], SPHINX [31], Google [35], Mercator [32] and Salticus [13].

Some crawlers released under the GNU public license include [5] and [23]. They have less features than their commercial counterparts, but are very fast. For commercial products, see [4].

Other studies focus on parallelization [17, 37], discovery and control of crawlers [39, 38, 3], accessing contents behind forms (the “hidden” Web) [36], mirror detection [18], freshness [19, 22], URL ordering [33, 21], focused crawling [24], and characteristics of the Web that are relevant to the crawler performance such as server response time [30], Web page changes [16, 25, 11], Web structure [12, 8], and how Web servers can help crawlers [10]

3 Software Architecture

The original architecture for this crawler was presented in [15], but it has experienced some changes. The system is designed to be:

- *Fast*: all of its data structures are specially designed to be fast, accounting for the specific access patterns of the crawler.
- *Robust*: it is designed as a modular set of very specific tasks. For instance, the program that does the network input/output is different from the program that parses pages.
- *Smart*: the crawler has access to all the information about pages, so it can take better decisions about which pages to visit next.
- *Flexible*: most of the parameters, including the score function, are configurable.

The main modules of the crawler are depicted in Figure 1:

- *Robot Seeder*: this module adds URLs to a structure specially designed to be fast in determining if a Web page has been seen before.
- *Robot Manager*: this module calculates scores and freshness for each page, and determines which pages must be crawled or re-crawled in the next batch.
- *Robot Harvester*: this module fetches pages from the Web. It can create a short-term schedule to make efficient use of the network resources available and to avoid overloading Web sites. Many harvesters can be running in different machines with one manager, as in [32].
- *Robot Gatherer*: this module parses the contents of Web pages, creates summaries of them, and passes the discovered URLs to the seeder program.

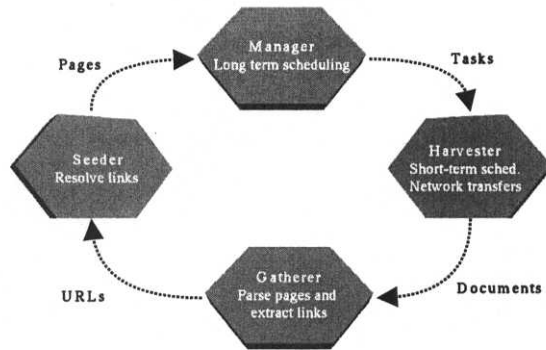


Figure 1: Modules of this Web crawler.

Some important features of the crawler we have designed and implemented are:

- Entirely written in C++.
- Uses a three-level cache for IP addresses.
- Implements pagerank, and multiple ways of calculating the priority for each page.
- Detects slows sites and re-schedules them.
- HTML parser to keep mainly the structural tags, removing most of the formatting tags.
- Duplicate detections by text analysis.
- Completely configurable with a configuration file in XML, using LibXML2 [2], including maximum depth for static and dynamic pages and weights in score calculation.
- Robot exclusion protocol [29], with extra time between requests for smaller sites.

4 Long-term Scheduling

This crawler relies on assigning a score (between 0 and 1) to each page; the higher the score, the better the page. The score summarizes the known metadata about a Web page in a single number.

Useful score functions include evidence about:

- *Link analysis measures*: pagerank [34], hubs and authorities [28], backlink count, or another measure of the link popularity of this page [20].
- *Text analysis*: the textual content can be compared with a query or a set of queries that try to define a category of pages, as in focused crawlers.

- *Access patterns*: if the logs of a search engine can be used, then the crawler can know which pages are more important for the users.
- *Site analysis*: all of the above, applied to the site where the page belongs. Also, pages belonging to fast Websites can be considered more important than those belonging to slower sites.
- Other factors.

The score function $score(p)$ and an estimation of the freshness, or probability that a page in the index is up-to-date, $freshness(p)$, are used to compute the value of a Web page in the index.

$$value(p) = score(p) \times freshness(p)^{1/\alpha} \quad (1)$$

The parameter α can be used to make the crawler re-visit pages more often. We use $\alpha = 1$ as default value.

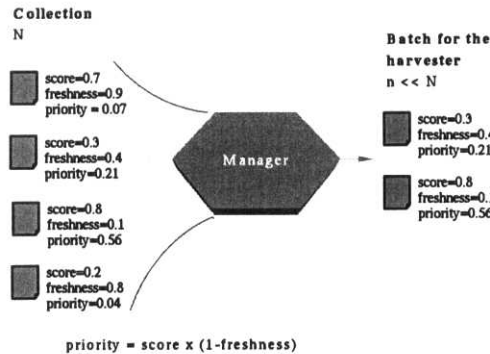


Figure 2: The process of creating a batch involves calculating estimators of freshness and score for each page.

The long-term scheduling algorithm we use generates batches of documents to be visited. In each round, we want to increase the total value of the index; every time we visit a page, its freshness is 1, and its value equals its score. Freshness decreases as time goes by if the page is not revisited. We use equation 1 to create a set of pages that will increase the value of the index more, as shown in Figure 2 the gain in value when bringing a fresh page p is $score(p) \times (1 - freshness(p))$ for $\alpha = 1$. Many batches can be created at the same time, because this can be an expensive calculation.

5 Politeness

The crawler is designed to be polite with Web sites. It only opens one connection at a time to a Website, it waits by default 30 seconds between accesses, it uses the robot exclusion protocol for sitewise limits and the robots meta-tag for pagewise limits. Also, it uses HTTP

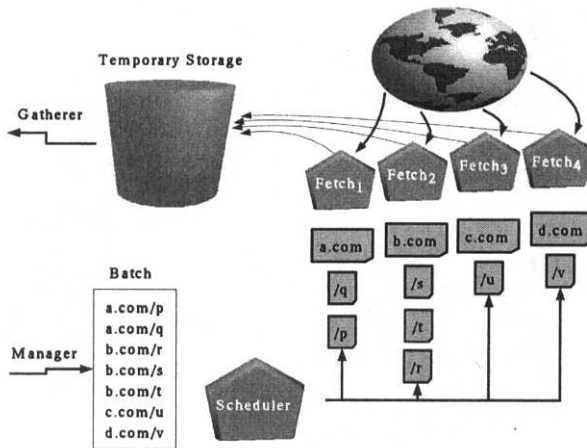


Figure 3: The harvester has a scheduler that creates a series of queues, one for each fetcher thread. At the end of the batch, a temporary storage is passed to the gatherer, that adds parses and adds pages to the main collection

if-modified-since headers to avoid downloading pages with no changes. Each site is assigned to a single harvester and to a single thread in the harvester, as shown in Figure 3.

The distribution of pages among Web sites on the Web is very skewed; its common for a vertical Web crawler to spend the last days of an entire crawl in a few Web sites. Based on that, we have implemented the crawler so it accesses pages from larger sites more often than smaller sites.

Mirrors on the Web are a significant challenge to Web crawlers, because a replicated collection usually has hundred or thousands of pages [18]. We compute a hash of the content of every downloaded page, and the crawler do not visit duplicates. The hashes are computed over the parsed contents (in which most of the tags are structural, and only a few formatting tags are allowed) so a page with minor stylistic changes is also considered as a mirror. The process of storing the text of a page, which is done by the gatherer, is shown in Figure 4.

6 Performance Issues

One of the major performance bottlenecks is DNS resolving. This has led some designs to include DNS resolvers as a standard part of the crawler [37]. In our case, we use the ADNS library [27] for asynchronous DNS resolving, and a three-level cache of IP address, using DnsCache [9], the memory in the process of the harvester, and the disk, because we save the IP address along with the metadata.

Another critical process is checking the links of visited pages to verify if they have been seen before. This is done with two hash-tables, one for the site names, and one for the paths, as shown in Figure 5; this is done to exploit locality of references in Web sites (most of the references on a page point to the same Web site). This global id solution is split into site ids and local file ids when storing the link graph of the Web to reduce the memory usage due to the locality of reference.

We have done our own implementation of a lightweight HTTP/1.1 [1] client, and we start

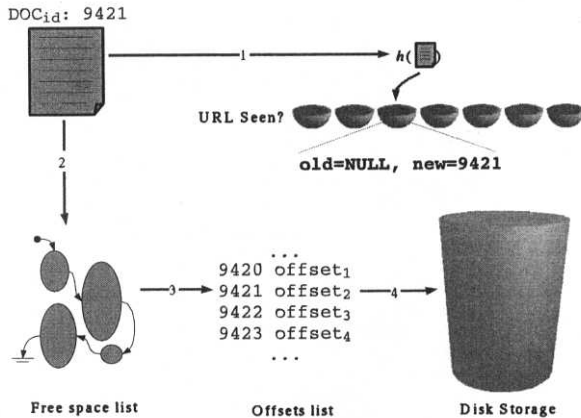


Figure 4: The storage subsystem, used by the gatherer, has a structure to determine if a page has been seen.

parsing pages only when the harvester is not running. This allows us to run up to 500 threads with a minimal impact in CPU load.

7 Final Remarks

Crawling is not as easy as using `wget` in mirror mode. To have a competitive performance, special software architectures that depend on networks, operating systems and programming environments must be designed. We have presented a crawler architecture that has evolved due to real experimentation that allows to find key issues that were not necessarily expected. For example, DNS resolving, dealing with large Web sites or storing the link graph of the Web.

There are four main areas in which we intend to do research in the short-term:

- Develop a framework to compare and optimize crawlers. This implies to answer questions like finding the optimal balance of processes and threads per CPU, the maximal number of threads to exploit a given bandwidth before saturation, or the optimal balance of processors and bandwidth.
- Web mining using this crawler; we intend to have a tool for gathering statistics about Web pages; we will use this to define a metric for evaluating different score functions and policies.
- Implement HTTP/1.1 pipelining for the largest Web sites.
- Develop and test a Web server plug-in to minimize the overhead due to request for non-changed pages, this plug-in can generate a lot of savings for both, the Web server and the Web crawler [10].

Acknowledgements: This work was funded by Millenium Nucleus Center for Web Research, Mideplan, Chile.

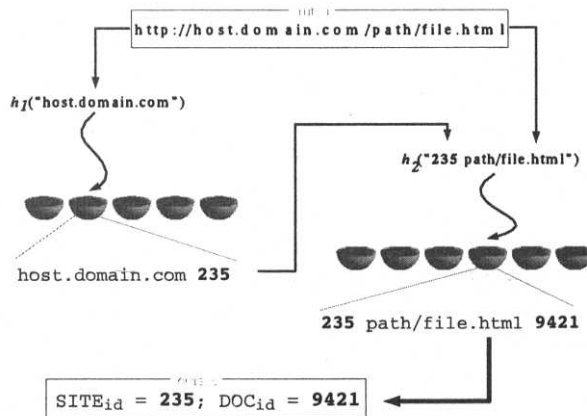


Figure 5: The URL index separates the URL in two parts; first is calculates a hash function of the host name, and then of the path. The most frequent operation is to resolve an unknown path in a known site.

References

- [1] HTTP - The hypertext transfer protocol, 1999.
- [2] LibXML - The XML C library for Gnome, 2002.
- [3] Robotcop - module to control robots on web servers, 2002.
- [4] Search engine watch. www.searchenginewatch.com/reports/, 2002.
- [5] Sebastien Ailleret. Larbin, a multi-purpose web crawler. larbin.sourceforge.net/index-eng.html, 2002.
- [6] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan. Searching the web. *ACM Transactions on Internet Technology*, 1(1):2–43, August 2001.
- [7] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley & ACM Press, Harlow, UK, 1999.
- [8] Ricardo Baeza-Yates and Carlos Castillo. Relating web characteristics with link based web page ranking. In *String Processing and Information Retrieval*. IEEE Cs. Press, 2001.
- [9] D. J. Bernstein. DNS cache - domain name system tools, 2002.
- [10] Onn Brandman, Junghoo Cho, Hector Garcia-Molina, and Narayanan Shivakumar. Crawler-friendly web servers. In *Proceedings of the Workshop on Performance and Architecture of Web Servers (PAWS)*, Santa Clara, California, 2000.
- [11] B. Brewington, G. Cybenko, R. Stata, K. Bharat, and F. Maghoul. How dynamic is the web? In *World Wide Web Conference*, 2000.
- [12] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, and A. Tomkins. Graph structure in the web: Experiments and models. In *9th World Wide Web Conference*, 2000.
- [13] Robin D. Burke. Salticus: guided crawling for personal digital libraries. In *ACM/IEEE Joint Conference on Digital Libraries*, pages 88–89, 2001.
- [14] M. Burner. Crawling towards eternity - building an archive of the world wide web. *Web Techniques*, 1997.
- [15] Carlos Castillo and Ricardo Baeza-Yates. A new crawling model. In *World Wide Web Conference*, Honolulu, USA, 2002. (Extended Poster).

- [16] J. Cho. The evolution of the web and implications for an incremental crawler. In *The VLDB Journal*, 2000.
- [17] J. Cho and H. Garcia-Molina. Parallel crawlers. In *11th International World-Wide Web Conference*, 2002.
- [18] J. Cho, N. Shivakumar, and H. Garcia-Molina. Finding replicated web collections. In *ACM International Conference on Management of Data (SIGMOD)*, 1999.
- [19] Junghoo Cho and Hector Garcia-Molina. Synchronizing a database to improve freshness. In *ACM International Conference on Management of Data (SIGMOD)*, pages 117–128, 2000.
- [20] Junghoo Cho, Hector Garcia-Molina, and Lawrence Page. Efficient crawling through URL ordering. *Computer Networks and ISDN Systems*, 30(1-7):161–172, 1998.
- [21] E. G. Coffman, Zhen Liu, and Richard R. Weber. Optimal robot scheduling for web search engines. Technical Report RR-3317, 1997.
- [22] Artur Czumaj, Ian Finch, Leszek Gasieniec, Alan Gibbons, Paul Leng, Wojciech Rytter, and Michele Zito. Efficient Web searching using temporal factors. *Theoretical Computer Science*, 262(1-2):569–582, 2001.
- [23] Lois Dacharay. Webbase web crawler. www.freesoftware.fsf.org/webbase/, 2002.
- [24] Michelangelo Diligenti, Frans Coetzee, Steve Lawrence, C. Lee Giles, and Marco Gori. Focused crawling using context graphs. In *26th International Conference on Very Large Databases, VLDB 2000*, pages 527–534, Cairo, Egypt, 10–14 September 2000.
- [25] Fred Douglass, Anja Feldmann, Balachander Krishnamurthy, and Jeffrey C. Mogul. Rate of change and other metrics: a live study of the world wide web. In *USENIX Symposium on Internet Technologies and Systems*, 1997.
- [26] D. Eichmann. The RBSE spider: balancing effective search against web load. In *1st World Wide Web Conference*, 1994.
- [27] Ian Jackson. ADNS - advanced, alternative, asynchronous resolver, 2002.
- [28] Jon Kleinberg. Authoritative sources in a hyperlinked environment. In *9th Symposium on discrete algorithms*, 1998.
- [29] Martijn Koster. Robots in the web: threat or treat ? In *ConneXions*, 4(4), 1999.
- [30] Binzhang Liu. Characterizing web response time. Technical report, Virginia State University, 1998.
- [31] R. Miller and K. Bharat. Sphinx: A framework for creating personal, site-specific web crawlers. In *7th World Wide Web Conference*, 1998.
- [32] M. Najork and A. Heydon. On high-performance web crawling, 2001.
- [33] Marc Najork and Janet L. Wiener. Breadth-First Crawling Yields High-Quality Pages. In *Proceedings of the 10th International World Wide Web Conference*, pages 114–118, Hong Kong, May 2001. Elsevier Science.
- [34] L. Page, S. Brin, R. Motwain, and T. Winograd. The pagerank citation algorithm: bringing order to the web. In *7th World Wide Web Conference*, 1998.
- [35] Lawrence Page and Sergei Brin. The anatomy of a large-scale hypertextual web search engine. In *7th World Wide Web Conference*, 1998.
- [36] Sriram Raghavan and Hector Garcia-Molina. Crawling the hidden web. In *Proceedings of the Twenty-seventh International Conference on Very Large Databases*, 2001.
- [37] Vladislav Shkapenyuk and Torsten Suel. Design and implementation of a high-performance distributed web crawler. In *ICDE*, 2002.
- [38] Jerome Talim, Zhen Liu, Philippe Nain, and Edward G. Coffman Jr. Controlling the robots of web search engines. In *SIGMETRICS/Performance*, pages 236–244, 2001.
- [39] P.N. Tan and V. Kumar. Discovery of web robots session based on their navigational patterns. *Data Mining and Knowledge discovery*, 2001.

Learnable Topic-specific Web Crawler

Niran Angkawattanawit and Arnon Rungsawang
Massive Information & Knowledge Engineering
Department of Computer Engineering, Faculty of Engineering
Kasetsart University, Bangkok 10900 Thailand.
Email: {g4365015, fengannr}@ku.ac.th

Abstract. Topic-specific web crawler collects relevant web pages of interested topics from the Internet. There are many previous researches focusing on algorithms of web page crawling. The main purpose of those algorithms is to gather as many relevant web pages as possible, and most of them only show the approaches of the first crawling. However, no one has ever mentioned some important questions, such as how the crawler does during the next crawlings, does the crawling process can be done in an incremental way, how to track the change of web pages, etc. In this paper, we present an algorithm that covers the detail of both the first and the consecutive crawlings. For efficient result of the next crawling, we keep the log of previous crawling to build some knowledge bases: starting URLs, topic keywords and URL prediction. These knowledge bases are used to build the experience of the topic-specific web crawler to produce better result for the next crawling.

1. Introduction

The rapid growth of the Internet has put us into trouble when we need to find information in such a large network of databases. At present, using the search engine becomes an easy way to seek the information. We may classify the search engines into two kinds by the topics of the information: topic-general search engine and topic-specific search engine. Topic-general search engine provides us the information in general topics. On the other hand, topic-specific search engine provides us the information in specific topics, such as the topics about traveling, sports, shopping, education, etc. The information gathering of the topic-general search engine uses an ordinary web crawler while the topic-specific search engine uses a topic-specific web crawler.

There are many previous researches mentioning about the topic-specific web crawling methods in the literature. For example, they are the best-first search [18], pagerank [1,2,3], content learning [4], context-graph learning [5], shark-search algorithm [7], adaptive agent [6,9,13], reinforcement learning [10], artificial life agent [17], arbitrary predicate [19], WTMS [11,12], etc. The main purpose of these algorithms is to gather the most relevant web pages as possible. However, the efficiency of a topic-specific web crawling is demonstrated as the proportion of the number of relevant web pages and the total number of downloaded web pages. If this proportion is high during early period of the crawling process, this means that the crawler can collect more relevant web pages than irrelevant ones. Although, these algorithms are very efficient, they only provide the detail of the first crawling. However, to freshen the data of the search engine, the crawling processes need to be done over and over. The problem we concentrate here is whether the next crawling process can be done in an incremental way, or how we can keep track the updating of web pages.

This paper presents an improved shark-search based algorithm that covers the detail of the first and the consecutive crawlings. We add a learning ability accumulated from

previous crawling to improve the efficiency of consecutive crawling processes. We argue that the crawler should have three kind of learnable knowledge as follows:

- *Starting URLs* – Because the crawler must collect as many relevant web pages as possible, it then needs a set of some good starting URLs which point to many relevant web pages.
- *Topic keywords* – Because the crawler must avoid collecting irrelevant pages, the crawler needs some appropriate keywords of interested topics for relevant page decision.
- *URL prediction* – Because the crawler must predict any URL in an undownloaded set whether it is a relevant web page, the crawler needs to predict those undownloaded pages according to the seen content of those pages from the previous crawling.

These knowledge bases are considered as the experience of the crawler. Therefore, a crawler which includes these knowledge bases should provide an efficient result during the next crawling.

This paper is structured as follows. Section 2 gives background knowledge such as the page similarity and the page importance needed to use in our crawling algorithm. Section 3 shows how to build knowledge bases for a crawler. Section 4 details our algorithm exposing the first crawling and the next crawling processes. In Section 5, we analyze the learning ability of a crawler, define the kind of the learning ability, and show how the consecutive crawls can be done more efficiently. Finally, Section 6 concludes the paper.

2. Background Knowledge

We need to include an analysis of web pages to a topic-specific web crawler because the crawler must collect relevant web pages before irrelevant ones. In this paper, the crawler uses *page similarity* to decide whether the pages are matched the interested topics and *page importance* to select more important pages.

2.1 Page Similarity

The analysis of page similarity is the most important thing for a topic-specific web crawling. Because the crawler is a program, it is not able to decide, like human decision, whether a web page is relevant or not. Consequently, the crawler must compute some values from web pages to basically determine the relevancy. We use similarity score for the decision of the crawler. At the beginning of the first crawling, the crawler has not seen the content of starting URLs until those pages are retrieved. The crawler must then collect the pages of all starting URLs to see their content and extract new URLs. To crawl new URLs, the crawler should predict their relevancy. Normally, most web editors often include some links which relate to page contents. Thus, the crawler can be guided by similarity score of the parent page and the anchor text appearing in the parent page. The crawler uses both combination score of parent page similarity and anchor text similarity to predict whether a new URL should be crawled. If the score of that new URL is low, the crawler will not retrieve it since it will not be useful.

There are many ways to compute the page similarity, but we here only focus on the *vector space* [8] approach. The vector space is widely used in many information retrieval researches. The idea of the vector space is to imagine a web page as a vector. Each distinct word in that page is considered to be an axis of a vector in a space. The direction of a vector characterizes the content of a web page. Two web pages are assumed to be relevant, i.e. talking with the same topic, if their vectors point to the same direction. Mathematically, if two vectors point to the same direction, this means that they take an angle of zero degree.

Cosine value of zero degree angle is 1. Therefore, we define $\text{sim}(q, d)$ to be the page similarity function with the formula demonstrated in Equation 1 below.

$$\text{sim}(q, d) = \frac{\sum_{k \in q \cap d} f_{kq} f_{kd}}{\sqrt{\sum_{k \in d} f_{kd}^2} \sqrt{\sum_{k \in q} f_{kq}^2}} \quad (1)$$

Here, q is a topic keyword set of an interested topic, d is the web page which we want to compare. f_{kq} and f_{kd} are the member of term k in query q and web page d , respectively. The range of $\text{sim}(q, d)$ value is between 0 and 1, and the similarity increases as this value increases.

2.2 Page Importance

We may classify the analysis of the page importance into two kinds by the importance of attributes: topic importance and link importance. Topic importance page is a page which relates to a topic. We may compute topic importance using the formula of page similarity in Equation 1. Link importance page is a page which is pointed from many other pages, in other words, many other pages concede this web page. There are many popular algorithms to calculate link importance such as PageRank [2,3], HITS [14,15], and BHITS [16]. Although, the internal detail of these algorithms are different, we do not discuss all of them here. We only focus on BHITS because it is an important key of our crawling algorithm (see Section 4).

BHITS [16] is an algorithm improved from HITS [14,15]. This algorithm computes two scores for each web page: a *hub* score and an *authority* score. Pages that have high authority scores are expected to have relevant content, whereas pages with high hub scores are expected to contain *links* to relevant pages. The intuition is as follows. A page which points to many other pages is a good hub, and a page that many other pages point to is a good authority. Transitively, a page that points to many good authorities is an even better hub, and similarly a page pointed to by many good hubs is an even better authority. Hubs and authorities exhibit what could be called a *mutually reinforcing relationship* [14]. However, there are some problems of mutually reinforcing relationship. Sometimes, a set of pages on one host point to a single page on the second host. This drives up the hub scores of the pages on the first host and the authority score of the page on the second host. The reverse case, where there is one page on the first host pointing to multiple pages on the second host, creates the same problem. Another problem is that the pages are not relevant to the topic but they have either high hub scores or high authority scores. This is called the *topic drift* problem [16]: the most-highly ranked authorities and hubs tend not to be about an interested topic. However, these problems have mentioned and solved by BHITS algorithm. For the problem of mutually reinforcing relationship between hosts, we will give fractional weights to links in such cases: if there are k links from pages on the first host to a single page on the second host we give each link an *authority weight* of $1/k$. This weight is used when computing the authority score of the page on the second host. If there are l links from a single page on the first host to a set of pages on the second host, we give each link a *hub weight* of $1/l$. In Section 3, we will show why and how we use BHITS algorithm.

3. Knowledge Bases of the Web Crawler

When we want to do something, we must learn and practice to improve our skill. We perhaps need suggestion about where the thing is, and where we can get it from the information service. Similarly, when a crawler has a task of crawling to do, it also needs some suggestion, i.e. the set of starting URLs, from the search engine about where relevant web pages are. After the crawler downloads a page, it will extract a set of URLs pointing to

many other pages. If the crawler has to decide whether each URL should be retrieved or not, it must analyze the content of their parent pages whether they relate to the topic of interest. Like us human, when we arrive at a place where the thing is, such as a shop, we can see all products in the shop and perhaps the product we want. If we ask the seller about the other shops, the answer is believable or not depending on the quality of the shop. However, to find a source of product is not easy if we cannot explain clearly the characteristic of the product we need, or perhaps we use a non well-known word to describe that product. Comparing with the crawler, topic keywords are important to reach the relevant web pages. If the keywords are too general or related to many other topics, web pages being collected may not be relevant. For a set of extracted URLs of the first crawling, the crawler will predict whether each URL is relevant or not. But, it is difficult to do that when the crawler has never seen those pages before. Like us, if we had never gone to the shop, we perhaps do not know all products in the shop and what the product we want look likes.

When we have some experience already, we can choose the first and the next shop to go. We can also choose by the number of products in the shop or a good advice from the seller. For a non well-known word we have ever used to call the product, we can choose it better after gaining more experience. We can also guess whether there is a product in a shop or not though we have never been in that shop before. In the same way, using the knowledge bases the crawler will be able choose the starting URLs pointing to many other relevant pages during the consecutive crawlings. Moreover, the crawler also keeps good topic keywords when it visits a web page, it will be able to predict whether a URL is relevant or not as it has ever been known the content of that URL before from the past crawling process.

For our web crawler, we build some knowledge bases from the previous crawling. These knowledge bases are: *starting URLs*, *topic keywords*, and *URL prediction*. Although, previous crawling may not be efficient, next crawling may be more efficient when the crawler uses knowledge bases as its experience. Starting URLs support the crawler to collect as many relevant web pages as possible. Topic keywords support the crawler to recognize appropriate keywords matching the topic. URL prediction supports the crawler to predict the relevancy of the pages of unvisited URLs.

3.1 Starting URLs

Starting URLs are very important for the crawler. If these URLs do not contain relevant web pages, the crawler has less possibility to find many other relevant ones. While the crawler does not have any knowledge base, the starting URLs for an interested topic can be deduced from a search result of a selected search engine. Normally, search engines will rank their search results by some popular ranking techniques such as PageRank [1,2,3], HITS [14,15], or BHITS [16], using authority and hub scores. If a page has the highest authority score, it will be ranked to the first result. Therefore, this set of starting URLs deduced from the selected search engines are the pages of high authority scores and are probably relevant pages. However, some of these starting URLs may have low hub scores, which may makes the crawler not to be able to find many others relevant pages.

To solve the above problem, we in general want a set of starting URLs in which each URL has both a high hub score, and a high authority score. A good hub page will point to many relevant pages and many hosts. If a page point to many other pages on the same host, we give it a hub weight (explained in Section 2.2). Consequently, to build the knowledge base of the starting URLs, we need a number of web pages from the previous crawling to compute hub and authority scores of pages using BHITS algorithm. Before we begin the computation, we must first analyze the previous crawling pages using the following algorithm:

- (1) Let R be the set of relevant pages.
- (2) Let N be the set of pages on which we focus.
- (3) Set $N := R$
- (4) For every page n in R ,
 - (5) Let $F(n)$ be the set of all pages n points to.
 - (6) Let $B(n)$ be the set of all pages pointing to n .
 - (7) Add all pages in $F(n)$ to N .
 - (8) If $|B(n)| \leq d$, then
 - (9) Add all pages in $B(n)$ to N .
 - (10) Else
 - (11) Add an arbitrary set of d pages from $B(n)$ to N .

The number of members in $B(n)$ may to be hundreds or thousands, therefore, we here let the threshold d to limit the number of members in N . When N have been constructed, we will calculate hub score and authority score using the following BHITS algorithm:

- (1) Let N be the set of pages on which we focus.
- (2) For every page n in N ,
 - (3) Let $H[n]$ be hub value of page n .
 - (4) Let $A[n]$ be authority value of page n .
 - (5) Initialize $H[n]$ and $A[n]$ to 1.
- (6) While the vectors H and A have not converged:
 - (7) For all n in N ,
 - (8) $A[n] := \sum_{(n',n) \in N} H[n'] \times \text{auth_wt}(n',n) \times \text{sim}(q,n')$;
 - (9) $H[n] := \sum_{(n,n') \in N} A[n'] \times \text{hub_wt}(n,n') \times \text{sim}(q,n')$;
 - (10) Normalize the H and A vectors.

H and A vectors are hub and authority scores of all pages n in N . In line 10, we normalize H and A vectors to be one-unit vectors. This means that the range of hub and authority scores are within 0 and 1. We let (n,n') be the page n pointing to page n' and (n',n) be the page n' pointed from page n . For auth_wt , this is an authority weight. If there are k links from pages in the first host pointing to a page in the second host, we give each link an $\text{auth_wt}(n',n)$ of $1/k$. This weight is used when computing the authority score of web pages on the second host. And, $\text{hub_wt}(n,n')$ is an hub weight. If there are l links pointed from a page in the first host to a set of pages in the second host, we give each link a $\text{hub_wt}(n,n')$ of $1/l$. Finally, $\text{sim}(q,n')$ is the similarity score between page n' and topic q . We use similarity score to solve the problem of topic drift.

3.2 Topic Keywords

Normally, a document having its content related to a topic is composed of a set of keywords which frequently appear in the topic. For example, a tcp/ip document is often contained by the following keywords: “tcp, ip, header, packet” or “protocol” etc. The crawler uses these keywords as a clue or guideline in topic-specific crawling. Sometimes, the crawler misses some relevant pages because the crawler has too little significant keywords or has not enough popular keywords of the topic under considerations. For example, the crawler does the topic-specific crawling of tcp/ip documents using only the following keywords: “tcp” and “ip”, this means that any page containing the words tcp or ip will be retrieved by the crawler. On the other hand, any page which does not contain the words tcp or ip will not be retrieved although it contains the words “header, packet” or “protocol”. Therefore, the topic keywords is an important factor and should be built as a knowledge base for the crawler.

To build the knowledge base of topic keywords, we need a set of web pages. We will extract some texts within the <Title></Title> tag and the <A> tag from each relevant page. Therefore, for a set of pages related to an interested topic, denoted N , title texts extracted from N give the crawler some keywords of the topic. The crawler will memorize

these keywords as the most popular keywords in the topic. For text within the $\langle A \rangle \langle /A \rangle$ tag, we extract the anchor text of each link in N , which each URL link points to other pages. We use the anchor text as topic keywords because the crawler will learn which keywords frequently used in hub pages. The step to build knowledge base of topic keywords is as the following:

- (1) Let N be the set of pages on which we focus.
- (2) For every page n in N ,
- (3) $Add_topic_keywords(q, n.title);$
- (4) For all (n', n) in N ,
- (5) $Add_topic_keywords(q, n'.link_to(n).anchor_text);$

Where $n'.link_to(n).anchor_text$ here is the anchor text of a link in page n' pointing to page n . Although, we should collect as many topic keywords as possible, we may not use all of them. Since some keywords with low occurrence frequency are less significant, we only select the keywords whose occurrence frequency is high enough to be topic keywords.

3.3 URL Prediction

During the first crawling, content of a web page is known when the crawler has finished downloading that page. While the page is not downloaded, the crawler should be guided by anchor text to determine whether the page should be retrieved. Sometimes the anchor text does not relate to the topic we are interested in, but the content of the page is relevant. On the contrary, the anchor text does relate to the topic, but the content of the page is not relevant. In other words, the crawler is blind during the first crawling. When comparing with human learning ability, before human have some skills and knowledge, they have never had any of them. Human will learn some experience to make skill and knowledge. Thus, we need our crawler to have an ability to learn something as human. Although, the crawler is blind during the first crawling, it can gain some experience by memorizing what it has seen. The benefit of content recognition will then be utilized in the next crawling. When the crawler must decide whether a URL contains a relevant page, it will get the previous seen content of the page from its database to predict the new content. We refer to this technique as the *URL prediction*. URL prediction helps the crawler to strongly decide whether a URL should be downloaded. To predict the content of a URL, we compute the page similarity between the content of the previous crawled page and the interested topic using the Equation 1. However, the content of pages may be changed, we then give a correcting factor (line 14 in Figure 2) to a score of the prediction. The detail of how to use the URL prediction will be explained in the Section 4.

4. Crawling Algorithms

The algorithm of our learnable web crawler is inspired from the shark-search algorithm described in [7] and from our previous research [17]. We may separate our crawling algorithm into two parts: crawling with no knowledge bases and crawling with knowledge bases. Crawling with no knowledge bases is used for the first crawling since the crawler has not had any knowledge yet. Crawling with knowledge base is used in the consecutive crawlings. These knowledge bases are continuously built from the previous crawlings.

In Figure 1, to begin the crawling with no knowledge base, we assign a topic to the crawler. The crawler gets t starting URLs from the search result of a selected search engine. The crawler will then rank the starting URLs using the similarity scores of each title and description text. Since there is no knowledge base, a topic keyword will then be used as the name of the topic. When the crawler gets a set of new URLs, it has to predict whether those URLs contain relevant pages or not. This can be done by computing two scores: the similarity score of the anchor text and the similarity score of the parent page. Considerably,

if the crawler does not have any knowledge, its crawling process may not be efficient. Therefore, we suppose that the next crawling should be better since the crawler will gain more experience from the first crawling. We denote *KB* to be the knowledge bases of the crawler.

```

1  Crawling_with_no_KB (topic) {
2    starting_urls := Search (topic, t);
3    keywords := topic;
4    foreach url (starting_urls) {
5      url_topic := url.title + url.description;
6      url_score := sim (keywords, url_topic);
7      enqueue (url_queue, url, url_score);
8    }
9    while (#url (url_queue) > 0) {
10     url := dequeue_url_with_max_score (url_queue);
11     page := fetch_new_document (url);
12     page_score := sim (keywords, page);
13     foreach link (extract_urls (page)) {
14       link_score :=  $\alpha \cdot \text{sim}(\text{keywords}, \text{link.anchor\_text}) + (1-\alpha) \cdot \text{page\_score}$ ;
15       enqueue (url_queue, link, link_score);
16     }
17   }
18 }

```

Figure 1. Pseudo-code of Crawling with no Knowledge base algorithm.

```

1  Crawling_with_KB (KB, topic) {
2    starting_urls := get_start_url_from_KB (KB, topic, t);
3    keywords := get_topic_keyword_from_KB (KB, topic);
4    foreach url (starting_urls) {
5      url_score := get_predict_score_from_KB (KB, topic, url);
6      enqueue (url_queue, url, url_score);
7    }
8    while (#url (url_queue) > 0) {
9      url := dequeue_url_with_max_score (url_queue);
10     page := fetch_new_document (url);
11     page_score := sim (keywords, page);
12     foreach link (extract_urls (page)) {
13       predict_link_score := get_predict_score_from_KB (KB, topic, url);
14       link_score :=  $\alpha \cdot (\beta \cdot \text{sim}(\text{keywords}, \text{link.anchor\_text}) + (1-\beta) \cdot \text{predict\_link\_score})$ 
+  $(1-\alpha) \cdot \text{page\_score}$ ;
15       enqueue (url_queue, link, link_score);
16     }
17   }
18 }

```

Figure 2. Pseudo-code of Crawling with Knowledge bases algorithm.

In Figure 2, to begin the crawling with knowledge bases, we assign a topic to the crawler. The crawler gets t starting URLs from the knowledge bases using the *get_start_URL_from_KB* function. These starting URLs are good hubs which point to many other relevant pages. Next, the crawler gets the keywords matching to the topic. Topic keywords are obtained from the *get_topic_keyword_from_KB* function. When the crawler gets a set of new URLs, it has to predict whether those URLs contain relevant pages or not by computing three scores: the similarity score of the anchor text, the similarity score of the parent page, and the predicting score of the URLs. The crawler computes the predicting score using the *get_predict_score_from_KB* function. For the consecutive crawling, the crawler should use this crawling with knowledge bases algorithm as it will be more efficient.

5. Analysis of Learnable Crawlers

A good crawler should collect many relevant pages during the beginning of the crawling process. We can evaluate the crawling efficiency using a *relevant graph* shown in Figure 3 below. The relevant graph records the relationship between the number of collected pages

and the number of relevant pages of each crawling. The x-axis is the percentage of the collected pages, and the y-axis is the percentage of the relevant pages.

We call three curves in Figure 3, the *learning curves*. The curve No.1 shows a bad crawling example since during the beginning of the crawling process the crawler can retrieve less relevant pages comparing to the total pages being collected. The curve No.2 is an assuming crawling example that we will represent it as a base line to separate between the good crawling example and the bad one. Finally, the curve No.3 is a good crawling example since during the beginning of the crawling process the crawler can retrieve most of the relevant pages comparing to the total pages being collected.

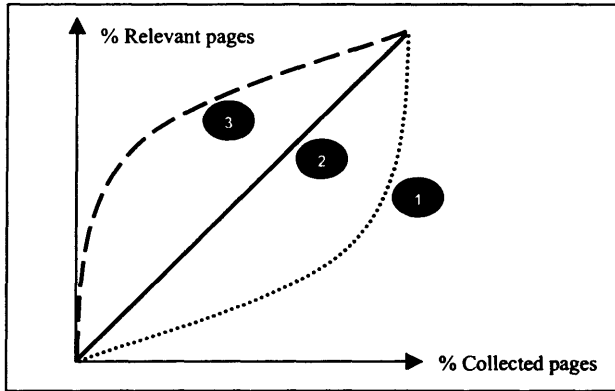


Figure 3. Learning Curves.

5.1 Types of Learning

We can see the learning tendency of a crawler during consecutive crawling by examining its learning curves. We define two types of learning: 1) Negative Learning and 2) Positive Learning.

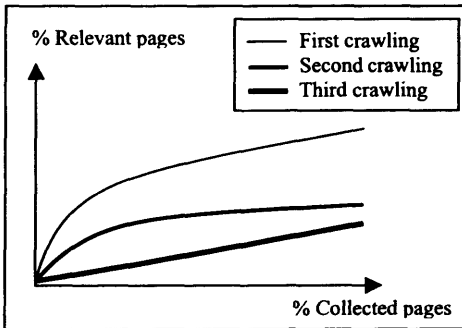


Figure 4. Negative Learning.

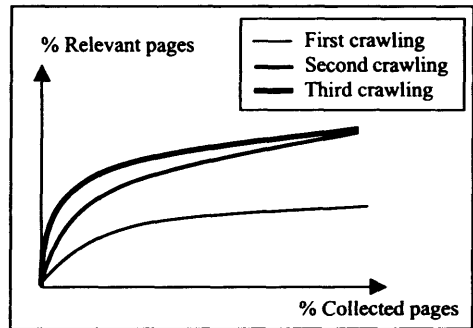


Figure 5. Positive Learning.

A crawler has negative learning ability when it can collect less relevant pages at the beginning of the crawling process during the consecutive crawling. For example, in Figure 4, the percentage of relevant pages the crawler can retrieve decreases from the first, the second and the third crawling, respectively. Evidently, this is not what we want the crawler to do. However, this can be happened when the content of previous collected pages is changed from relevant to irrelevant one, and the fault URL prediction may result the crawler in missing many relevant pages.

On the other way, a crawler has positive learning ability when it can collect much more relevant pages at the beginning of the crawling process during the consecutive crawling.

For example, in Figure 5, the percentage of relevant pages the crawler can retrieve increases from the first, the second and the third crawling, respectively. This result can be come from using the starting URLs, good topic keywords, and the fact that the URL prediction works as it should be. Good starting URLs should be the best hubs which point to many other relevant pages, so that the crawler has a high probability to collect many relevant pages during the beginning of the crawling process. Good topic keywords are those words that are more specific within a certain topic. This makes the crawler knows which pages are relevant or not. Finally, good URL prediction can results the crawler in collecting the pages having irrelevant anchor text but relevant content.

5.2 Efficient Learning

To reduce the network bandwidth usage, an efficient crawling of a topic-specific web crawler is preferred. A crawler has efficient learning ability if it can collect many relevant pages at the beginning of the crawling processes during the consecutive crawlings. Figure 6 shows an example of an efficient learning crawler. The tendency of the learning ability, illustrated by its learning curves, increases from the first till the fifth and the sixth crawling, respectively. The crawler can retrieve more than 90% of relevant pages when it visits less than 25% of the total pages collected from the first crawling.

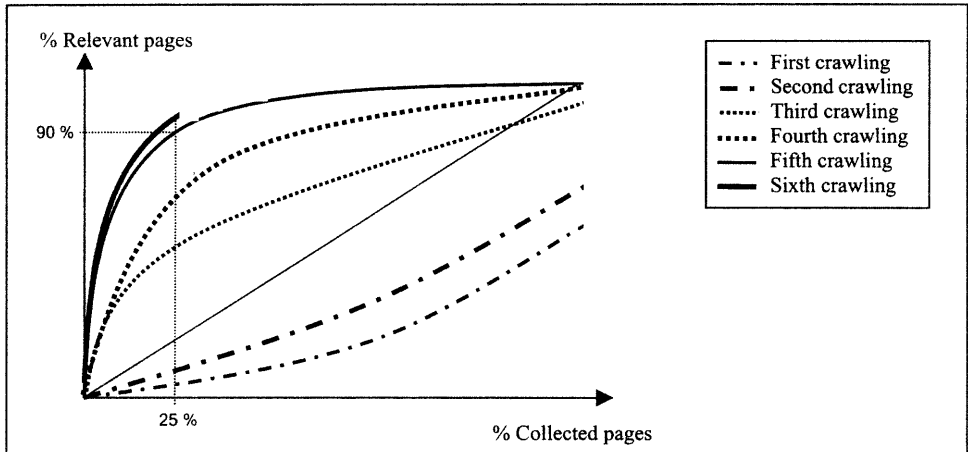


Figure 6. Efficient Crawling.

6. Conclusion

This paper presents a learnable topic-specific web crawler. There are three main parts discussed in the paper: crawler's knowledge bases, crawling algorithm, and analysis of crawler learning ability. The knowledge bases of the crawler are incrementally built from the log of previous crawling. For efficient result of the next crawling, we present three knowledge bases: starting URLs, topic keywords, and URL prediction. Good starting URLs support the crawler to collect as many relevant web pages as possible. Good topic keywords support the crawler to recognize appropriate keywords matching the required topic. Good URL prediction supports the crawler to predict the relevancy of the content of unvisited URLs. Crawling algorithm has been separated into two parts: crawling with no knowledge base and crawling with knowledge bases. Crawling with no knowledge base is used in the first crawling for Internet exploration. The information gathered from the first crawling has been accumulated to be the experience of the crawler, i.e. the knowledge bases, during the

consecutive crawling. Crawling with knowledge bases should be used in consecutive crawlings for more efficient result and better network bandwidth utilization.

References

- [1] J. Cho, H. Garcia-Molina, and L. Page. "Efficient Crawling Through URL Ordering." In *Proceedings of the 7th International World-Wide Web Conference*, 1998.
- [2] L. Page, S. Brin, R. Motwani, and T. Winograd. "The PageRank Citation Ranking: Bringing Order to the Web." *Stanford Digital Libraries Working Paper*, 1997.
- [3] S. Brin and L. Page. "The anatomy of a large-scale hypertextual web search engine." In *Proceedings of the 7th International World Wide Web Conference*, 1998.
- [4] S. Chakrabarti, M. van den Berg, and B. Dom. "Focused crawling: a new approach to topic-specific Web resource discovery." In *Proceedings of the 8th International World Wide Web Conference*, 1999.
- [5] M. Diligenti, F.M. Coetzee, S. Lawrence, C.L. Giles, and M. Gori. "Focused Crawling Using Context Graphs." In *Proceedings of the 26th International Conference on Very Large Data Bases*, 2000.
- [6] M. Degeratu, G. Pant, and F. Menczer. "Latency-Dependent Fitness in Evolutionary Multithreaded Web Agents." In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2001)*, *EcoMAS Workshop*, 2001.
- [7] M. Hersovici, M. Jacovi, Y.S. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur. "The Shark-Search Algorithm – An Application: Tailored Web Site Mapping." In *Proceedings of the 7th International World Wide Web Conference*, 1998.
- [8] R. Baeza-Yates, and B. Ribeiro-Neto. "Modern Information Retrieval." *Addison-Wesley*, pp. 27-28, 1999.
- [9] F. Menczer, and A.E. Monge. "Scalable Web Search by Adaptive Online Agents: An InfoSpiders Case Study." In *Intelligent Information Agents*, Springer.
- [10] J. Rennie and A. McCallum. "Efficient Web Spidering with Reinforcement Learning." In *Proceedings of the 16th International Conference on Machine Learning*, 1999.
- [11] S. Mukherjee. "WTMS: A System for Collecting and Analyzing Topic-specific Web Information." In *Proceedings of the 9th International World Wide Web Conference*, 2000.
- [12] S. Mukherjee. "Organizing Topic-specific Web Information." In *Proceedings of the 11th ACM Conference on Hypertext*, 2000.
- [13] F. Menczer and R. Belew. "Adaptive retrieval agents: Internalizing local context and scaling up to the web." *Machine Learning*, 39(2/3):203-242, 2000.
- [14] J. Kleinberg. "Authoritative sources in a hyperlinked environment." In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1998.
- [15] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, and S. Rajagopalan. "Automatic resource compilation by analyzing hyperlink structure and associated text." In *Proceedings of the 7th International World Wide Web Conference*, 1998.
- [16] K. Bharat and M. Henzinger. "Improved Algorithms for Topic Distillation in a Hyperlinked Environment." In *Proceedings of the 21st ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998.
- [17] N. Angkawattanawit and A. Rungsawang. "Topic-specific Gathering using MIKE Spider." In *Proceedings of the 5th National Computer Science and Engineering Conference*, 2001.
- [18] M. Najork and I. N. Wiener. "Breadth-first search crawling yields high-quality pages." In *Proceedings of the 10th International World Wide Web Conference*, 2001.
- [19] C. C. Aggarwal, F. Al-Garawi, and P. S. Yu. "Intelligent Crawling on the World Wide Web with Arbitrary Predicates." In *Proceedings of the 10th International World Wide Web Conference*, 2001.

A Spatial Dimension for Searching the World Wide Web

M. Andrea Rodríguez Tastets

Department of Information Engineering and Computer Science

University of Concepción

and

Center for Web Research

University of Chile

Edmundo Larenas 215, Concepción, Chile.

andrea@udec.cl

Abstract. Few studies have explored spatial relations as they constrain the searching of documents in the World Wide Web (WWW). This paper presents the theoretical basis for a spatial searching of Web documents. It reviews spatial reasoning concepts associated with spatial relations and describes a model for organizing and deriving spatial relations based on a hierarchical structure of the space, that is, a conceptual model of the space in terms of connected regions. Using a study case, this work presents guidelines for how this model can be used for extending current searching techniques of Web documents to answer queries that are constrained by spatial relations.

1 Introduction

Several dimensions exist for searching information [1]. Traditionally, searching the WWW involves the occurrence analysis of keywords within textual documents. This type of searching continues being a very useful way to find information on a given topic. As users increasingly demand better and more complex information retrieval tasks, the technology needed to deal successfully with these issues must focus on several semantic dimensions underlying the text document. For example, a user may need information based on a temporal sequence or may need information of the relationship between different data sources or documents.

This work focuses on the spatial dimension of searching Web documents. In this context, we are interested in answering queries such as “find *hotels* in *Pucón, Chile*, and in *adjacent towns*.” Using current searching techniques, the users will need to specify a query with a disjunctive form of several conjunctive conditions, knowing all possible adjacent towns. This paper presents a knowledge-based approach to modeling spatial information based on a hierarchical structure of the space, that is, a semantic structure that organizes the space’s regions and these regions interrelations.

Usually, spatial information has been defined by any information that has a spatial dimension. Thus, all kind of graphic information has been cataloged as spatial. This work, however, deals with textual information that involves places or geographic regions, but ignores geometric information included in Web documents. An abstraction of the basic geometric properties (i.e. dimension and shape) of these places is handling in the

hierarchical structure of the space. The general idea behind this approach is that names of regions or places are mapped onto the semantic structure, such that, Web documents associated with these places do not need to incorporate geometric properties of the space within their texts. In this way, this work agrees with recent studies on information retrieval that have investigated the use of knowledge bases and similarity functions as a mechanism to compare terms [2, 3]. The general approach in these studies has been to map the local terms onto a shared ontology [4], in this work, the hierarchical structure of the space. An ontology captures a particular view of the world, supports intensional queries regarding the content of a database, defines semantics independently of data representation, and reflects the relevance of data without accessing them [5].

This paper considers that names of regions or places are spatial information, even when no geometry is involved, since they are typically used to refer spatial objects. In addition, even when positional information is often a basis for spatial components, other spatial components, such as spatial relations of adjacency and containment, do not require absolute positional data [6]. These spatial relations represent higher levels of abstraction than positional information [7], since they can be represented by qualitative primitives, such as connectivity [8], and with respect to a qualitative frame of reference (e.g., A is left to B) [9].

Spatial relations have obtained great attention by the research community; however, few studies have addressed access methods as they relate to queries based on spatial relations. Combining Geographic Information systems and conceptual structures as modeled in relational concept analysis, Priss and Old [10] proposed a new technique for information access. In this technique, linearly ordered information can be organized in conceptual structures and then, linked to spatial data that is handled by a geographic information system. Extending a previous work on conceptual, spatial, and temporal referencing of multimedia objects [11], Alani *et al.* [12] recently explored terminological systems for thematic and spatial access in digital library applications. They defined a semantic structure with associative, hierarchical and adjacency relations between name places and explored semantic distance measures that permit imprecise matching between query and data. This work follows closely the ideas by Alani *et al.*, but it differs in the level of abstraction used in the definition of the semantic structure. In this work, the semantic structure is a hierarchical subdivision of the space and the relations between regions are modeled by using spatial reasoning techniques in order to reduce the stored data and step up the search process.

A recent work by McCurley [1] uses a knowledge base of the space to search Web documents. Unlike McCurley's work, this work concentrates on the modeling aspects of the space and how this can be used for information retrieval purposes. It explores spatial reasoning techniques for further analysis of topological, cardinal and distance relations. This work, leaves for a future study aspects related to how names of places are mapped onto the semantic structure of the space, which was already treated by McCurley.

The organization of the paper is as follows. Section 2 describes concepts associated with reasoning about spatial relations. Section 3 presents the model of spatial regions and interrelations. A study case is described in Section 4. Conclusions and future research directions are presented in Section 5.

2 Reasoning about Spatial Relations

The term *spatial relations* refers to those relations between objects that involve their relative positions. Although spatial relations define constraints for retrieving data from data collections, they are not necessarily part of the explicitly stored data and, therefore,

methods for deriving them are indispensable. How spatial relations are derived lies in the area of spatial reasoning. Spatial reasoning is commonly used to make inferences about the spatial relations among objects using a subset of known spatial relations. A system with spatial reasoning capability for its data management and query language must consider an unambiguous formalization of individual spatial relations as well as combinations of spatial relations to generate plausible answers [13-15].

Spatial reasoning is often classified into quantitative and qualitative depending on the type of information used in the reasoning process. This work is intended to combine qualitative and quantitative reasoning in searching Web documents with a spatial dimension. Qualitative reasoning is the one that people perform regarding a particular spatial situation based on a finite set of distinguished spatial entities and on a finite set of spatial interrelations among these spatial entities [9]. This type of reasoning allows us to make predictions, diagnosis, and explain the behavior of systems when imprecise or incomplete information is available [16, 17].

Many studies in the area of spatial reasoning has been concerned with the construction of algebras to make inferences about spatial relations. Examples are compositions of topological relations [18, 19], cardinal directions [20, 21], approximate distances [22], and their combinations [9, 23, 24]. These studies follow the approach given for inferences about interval relations, i.e., they embed spatial relations within a space in which decisions about compositions can be derived. The result of these formalizations is a set of symbols (i.e., spatial relations) and a set of logical rules about the combinations of these symbols (i.e., compositions).

Traditionally, topology has been seen as the most abstract spatial structure, the one that is considered essentially qualitative [25]. Topological spatial relations are invariant under continuous transformations, such as rotation, translation, and scaling. Composition of topological relation have been previously studied [8, 18]. This work concentrates on a subset of the topological relations between regions—*disjoint*, *adjacent* or *meet*, *inside*, and *contains*—, since *non-disjoint* relations represent connectivity of regions and these relations are usually the type of topological relations that are used to express spatial interrelation between geographic objects [26]. For this subset of topological relations, the composition is given in Figure 1. Note that this subset of topological relations does not constitute a relation algebra, since it does not include the whole set of mutually exclusive spatial relations between regions.

























	 inside	 contains	 meets	 disjoint
 inside				
 contains				
 meets				
 disjoint				

Fig. 1. Partial composition table of topological relations between regions

Cardinality and orientation involve a reference object, a target object, and a fixed frame of reference to which the objects are related [21]. These relations are invariant under

continuous translation and scaling, but change with rotation. By considering cardinal direction and the perpendicular projection of the reference objects' minimum bounding rectangles (MBRs), the composition table is shown in Figure 2.

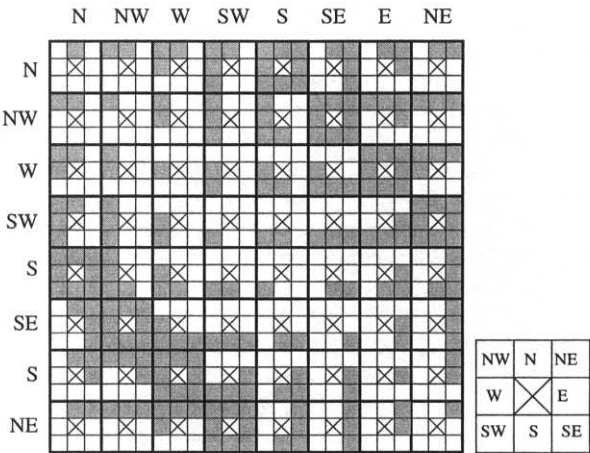


Fig. 2. Composition of cardinal directions

Qualitative distance relations correspond to a categorization of distance between two objects into qualitative symbols, such as near and far. An important aspect of the qualitative distance relations is their contextual dependencies [27]. Distance relations do not only depend on absolute distances, but also on the relative sizes, shapes, locations, task, and frames of reference. In order to model relative distance, a content measure that considers relative size of objects could be defined, such such as those measures defined in [28]. Combinations of spatial relations have been studied as well [13, 23]. These studies apply the composition operation over different types of spatial relations to provide further reasoning capabilities.

The inferences defined above through composition tables are basis for retrieving information whose constraints are expressed by predicates of topological, distance, and cardinal relations.

3 Modeling Regions and Spatial Relations

The model of geographic regions uses a hierarchical structure, a tree (Figure 3). From an ontological point of view, this work uses a region-based approach to specifying spatial relations [25]. The basic primitives of the model are *cells of irregular tessellation*, called *regions* in this paper. Aggregations of *regions* make *conceptual regions*. Thus, while regions are always connected, a conceptual region may not, such as the case of an island that belongs to a county or state.

The hierarchical structure groups non-overlapping *regions* using the *inside* relation, or conversely, the *contains* relation. *Regions* may be defined at different levels of abstraction so, what is a region at level *n* of the tree may be part of a connected region at level *n - 1*. A hierarchical representation of geographic space has been widely used as a data structure and has desirable computational properties for organizing and indexing data [29]. As a simplification of objects' geometry, minimum bounding rectangles (MBRs) are defined in order to subsequently define a distance measure.

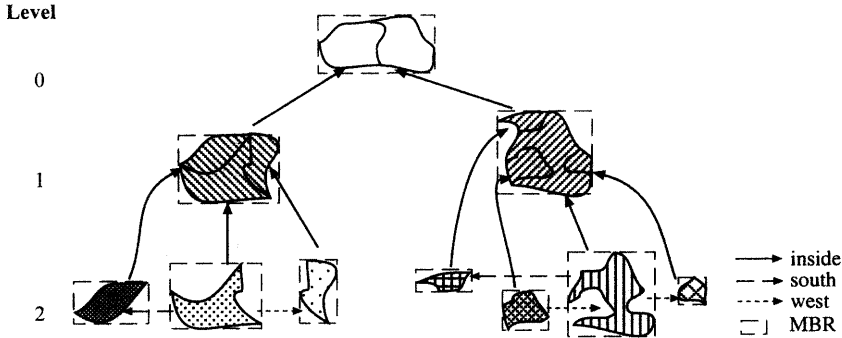


Fig. 3. Hierarchy of regions based on containment

From a conceptual data modeling point of view, the diagram of the static structure is shown in Figure 4.

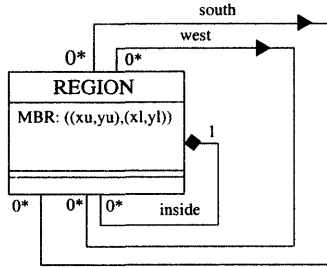


Fig. 4. Static structure diagram of a region

Although the *inside* or its converse *contains* relation is the principal relation of the model, additional spatial relations are *meet* or *adjacent*, and cardinal relations such as, *north*, *south*, *west*, *east*, *north-west*, *north-east*, *south-west*, and *south-east*. By axioms A1-A5 of converse relations and adjacency of cardinally connected regions, it is possible to reduce the number of relations stored in the database. Cardinal relations are only stored for adjacent regions. Likewise, only one direction of the spatial relations is stored, since the converse relation can be automatically derived.

$$(\forall r_1 \neq r_2) \text{inside}(r_1, r_2) \equiv \text{contains}(r_2, r_1) \quad (\text{A.1})$$

$$(\forall r_1 \neq r_2) \text{west}(r_1, r_2) \equiv \text{east}(r_2, r_1) \quad (\text{A.2})$$

$$(\forall r_1 \neq r_2) \text{north}(r_1, r_2) \equiv \text{south}(r_2, r_1) \quad (\text{A.3})$$

$$(\forall r_1 \neq r_2) \text{north_west}(r_1, r_2) \equiv \text{south_east}(r_2, r_1) \quad (\text{A.4})$$

$$(\forall r_1 \neq r_2) \text{meet}(r_1, r_2) \equiv \text{north}(r_1, r_2) \vee \text{south}(r_1, r_2) \vee \text{west}(r_1, r_2) \vee \text{east}(r_1, r_2) \vee \text{north_west}(r_1, r_2) \vee \text{south_west}(r_1, r_2) \vee \text{north_east}(r_1, r_2) \vee \text{south_east}(r_1, r_2) \quad (\text{A.5})$$

In the hierarchical structure, *cardinal* relations, and therefore, *meet* relations, are defined at the leaves of the tree. Then, it is possible to define an inference mechanism to derive relations between objects at different levels of the hierarchy. These inferences satisfy the following axioms (A6-A9), where r_i are regions at a level $n - 1$ and x_i are regions at level n :

$$(\forall r_1 \neq r_2) \text{meet}(r_1, r_2) \equiv (\exists x_1, x_2) (\text{meet}(x_1, x_2) \wedge \text{inside}(x_1, r_1) \wedge \text{inside}(x_2, r_2)) \quad (\text{A.6})$$

$$(\forall \eta \neq \eta_1) \text{meet}(\eta, \eta_1) \equiv (\exists x)(\text{meet}(\eta_1, x) \wedge \text{inside}(x, \eta)) \quad (\text{A.7})$$

$$(\forall \eta \neq \eta_2) \text{disjoin}(\eta, \eta_2) \equiv \neg \text{meet}(\eta, \eta_2) \quad (\text{A.8})$$

$$(\forall \eta \neq \eta_1) \text{disjoin}(\eta, \eta_1) \equiv \neg \text{meet}(\eta, \eta_1) \wedge \neg \text{inside}(\eta_1, \eta) \quad (\text{A.9})$$

Axioms can be applied recursively together with composition operations defined in previous Section. Thus, knowing the relations between regions at the leaves allows us to derive relations at all levels of the tree. At the conceptual level, it is also possible to define axioms for topological relations between *conceptual regions* c_i (Axioms 10-16). In this way, a more complete formalization of topological relations is handled by the geographic semantic structure.

$$(\forall c_1, c_2) \text{connected}(c_1, c_2) \equiv (\exists \eta, \eta_2)(\text{inside}(\eta, c_1) \wedge \text{inside}(\eta_2, c_2) \wedge \text{meet}(\eta, \eta_2)) \quad (\text{A.10})$$

$$(\forall c_1, c_2) \text{inside}(c_1, c_2) \equiv (\forall r)(\text{inside}(r, c_1) \supset \text{inside}(r, c_2)) \quad (\text{A.11})$$

$$(\forall c_1, c_2) \text{overlap}(c_1, c_2) \equiv (\exists r)(\text{inside}(r, c_1) \wedge \text{inside}(r, c_2)) \quad (\text{A.12})$$

$$(\forall c_1, c_2) \text{properOverlap}(c_1, c_2) \equiv \text{overlap}(c_1, c_2) \wedge \neg \text{inside}(c_1, c_2) \wedge \neg \text{inside}(c_2, c_1) \quad (\text{A.13})$$

$$(\forall c_1, c_2) \text{equal}(c_1, c_2) \equiv \text{inside}(c_1, c_2) \wedge \text{inside}(c_2, c_1) \quad (\text{A.14})$$

$$(\forall c_1, c_2) \text{meet}(c_1, c_2) \equiv \text{connected}(c_1, c_2) \wedge \neg \text{overlap}(c_1, c_2) \quad (\text{A.15})$$

$$(\forall c_1, c_2) \text{disjoin}(c_1, c_2) \equiv \neg \text{connected}(c_1, c_2) \quad (\text{A.16})$$

Following the same strategy used for topological relations, at the leaves of the tree cardinal relations need to be defined. Axioms 17 and 18 define the cardinal relation *north*, which is a model of the axioms for all cardinal relations.

$$(\forall \eta \neq \eta_2) \text{north}(\eta, \eta_2) \equiv (\exists x_1, x_2)(\text{north}(x_1, x_2) \wedge \text{inside}(x_1, \eta) \wedge \text{inside}(x_2, \eta_2)) \quad (\text{A.17})$$

$$(\forall \eta \neq \eta_1) \text{north}(\eta, \eta_1) \equiv (\exists x)(\text{north}(x_1, x) \wedge \text{inside}(x, \eta)) \quad (\text{A.18})$$

For distance relations and, based on a modification of a previous work [28], a content measure of relative distances is defined in Equation 1. The diagonal of the reference region normalizes the distance between regions' MBRs. This normalization makes the content measure an a-dimensional and asymmetric value that depends on regions' sizes.

$$D(\eta, \eta_2) = \frac{\max_distance(MBR(\eta), MBR(\eta_2))}{diagonal(\eta)}, D(\eta_2, \eta) = \frac{\max_distance(MBR(\eta), MBR(\eta_2))}{diagonal(\eta_2)} \quad (\text{Eq. 1})$$

The content measure of distances can be pre-calculated and assigned between regions of the same level in the tree; however, this may produce a dense interconnection, since between any two regions there is always a distance relation. In this case, distances between parts with respect to their wholes are considered equal to zero.

Following the same ideas used for topological and cardinal relations, distance can be assigned to regions at the leaves of the tree, in which case, the minimum distance between regions' parts determines the distance between regions at a higher level in the tree. Then, a relative distance is obtained by determining the distance between two regions normalized by the maximum distance between any two regions at the same level in the hierarchical structure.

4 Study Case: Chilean Tourist Information in the WWW

In order to illustrate how this model of the geographic space can be used with retrieval purposes in the WWW, this paper presents as a case of study the access of tourist information in Chile. The idea of this approach is to handle a geographic semantic structure

that divides Chile into non-overlapping regions. Thus, a natural hierarchical structure follows the administrative boundaries of regions. Depending on the level of detail needed in the retrieval system, it is possible to further subdivide regions into counties and town boundary (Figure 5).

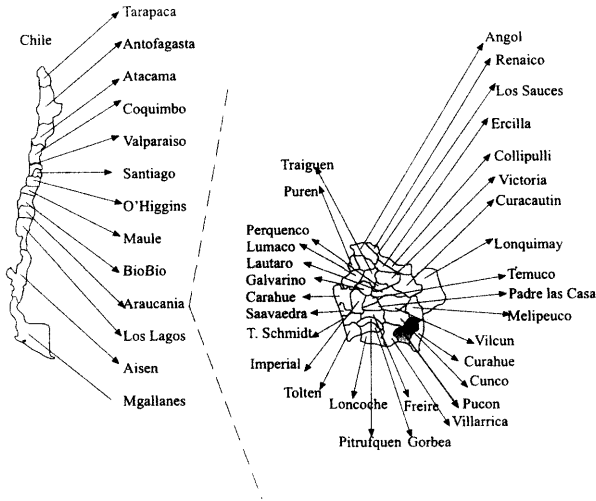


Fig. 5. Geographic Ontology of Chile

The static structure defined in Figure 4 can be translated into a representation using XML [30], whose DTD file is shown in Figure 6. This specification follows the schema of Thematic Maps [31], that is, a model based on topics and associations between topics. In this case, topics are regions and associations are spatial relations. This modeling strategy makes the geographic structure independent of the occurrence of Web documents.

```

<?xml version="1.0" encoding="UTF-8"?>
<ELEMENT Geographic_Hierarchy (region+, spatial_relation*)>
<ELEMENT region (baseName+, MBR?, occurrences*)>
<ELEMENT baseName (baseNameString)>
<ELEMENT baseNameString (#PCDATA)>
<ELEMENT MBR (upperLeft, lowerRight)>
<ELEMENT upperLeft (coordinate)>
<ELEMENT lowerRight (coordinate)>
<ELEMENT coordinate (X,Y)>
<ELEMENT X (#NUMBER)>
<ELEMENT Y (#NUMBER)>
<ELEMENT spatial_relation (distance+|cardinal+|topology+, targetObject)>
<ATTLIST region regionRef (#CDATA) regionTarg (#CDATA)>
<ELEMENT distance (#NUMBER)>
<ELEMENT cardinal ("north"|"west"|"north_west")>
<ELEMENT topology("in")>

```

Fig. 6. DTD of the Geographic Ontology

As an example, Figure 7 shows the specification of Pucón County, which is considered, in this case, a leaf of the geographic hierarchical structure.

```
<region>
  <baseName> <baseNameString>Pucón</baseNameString>
  <baseNameString>Pucon</baseNameString></baseName>
  <MBR>
    <upperLeft><coordinate> <X>325 </X> <Y>575 </Y></coordinate></upperLeft>
    <lowerRight><coordinate> <X> 334</X> <Y>564 </Y></coordinate></lowerRight>
  </MBR>
  <occurrence><resourceRef xlink:href="http://www.chi-hotels/antumalal.htm"/></occurrence>
  <occurrence>.....</occurrence>
</region>
<spatial relation><regionRef xlink:href="#Araucania"/>
  <topology> in </topology><targetObject xlinkRef="#Pucon"/>
</spatial relation>
<spatial relation><regionRef xlink:href="#Pucon"/>
  <cardinal> north </cardinal><regionTarg xlinkRef="#Melipeuco"/>
</spatial relation>
<spatial relation><regionRef xlink:href="#Pucon"/>
  <cardinal> north_west </cardinal><regionTar xlinkRef="#Cunco"/>
</spatial relation>
<spatial relation><regionRef xlink:href="#Pucon"/>
  <cardinal> west </cardinal><regionTart xlinkRef="#Villarica"/>
</spatial relation>
<spatial relation><regionRef xlink:href="#Curarrehue"/>
  <cardinal> north_west </cardinal><regionTart xlinkRef="#Pucon"/>
</spatial relation>
```

Fig. 7. Partial specification in XML of the space hierarchical structure

The hierarchical structure of the geographic space should be handled by the search engine, which needs to map geographic references in Web documents onto this hierarchical structure. For example, the Web document that describes different hotels in Pucón, such as the Web page of “Amtumalal Hotel” ([http:// www.chile-hotels/antumalal.html](http://www.chile-hotels/antumalal.html)), should be associated as an occurrence of Web documents in the county of Pucón. A traditional query can be “Hotels Pucón,” in which case, a ranking list of Web documents containing the keywords *hotels* and *Pucón* are displayed. This work proposes to incorporate spatial operators to be able to request “Hotels in Pucón” or “Hotels near to Pucón.” This type of query could retrieve a ranking set of Web documents based on a relative distance relation. In this case, Web documents with hotel information (obtained by classic web engines) would be ranked based on the relative distance between their geographic references and Pucón, which are mapped onto the geographic structure. The ranked relative distance is shown in Figure 8, where the counties surrounding Pucón are ranked from close (1) to far (10).

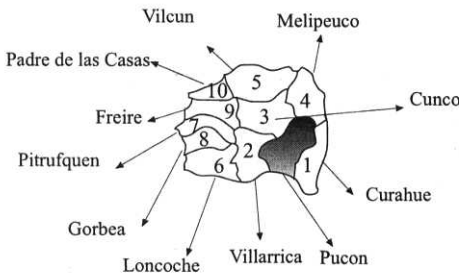


Fig. 8. Ranking of relative distance between closest counties and Pucón

Another query may be “ski centers *adjacent* to a lake.” To solve this kind of query, we should have added additional information to the space hierarchical structure containing

information of water bodies that are contained in each region. Then, Web documents whose geographic references are adjacent to regions with lakes would be retrieved. Thus, a spatial dimension for searching the WWW can be combined with any dimension capturing the semantic content of Web documents. This geographic reference can be associated with the highest detail in the geographic hierarchy (i.e., leaves of the tree) or with coarse descriptions at higher levels in the hierarchy. In such case, applying composition operations and axioms that were described in previous Section carries out the derivation of spatial relations.

5 Conclusions and Future Work

This paper has presented an approach for extending current WWW search capabilities to answering queries expressed by spatial operations (e.g., *in*, *north*, *west*, *near*, and so on). In this process, a hierarchical structure of the space was defined with axioms that describe topological, cardinal, and distance relations. This structure underlies the description of Web documents with geographic references given by places' names.

This work is in progress and much further investigation is needed. A more powerful tool should complement a spatial with a temporal or conceptual dimension. For example, the case study shows that based on a relative distance the closest county to Pucón is Curahue; however, from a tourist point of view, Villarrica is more similar to Pucón than Curahue. Such type of analysis needs spatial as well as other semantic information. In a full-implemented system, an efficient indexing schema will be needed for organizing and handling occurrences of Web resources associated with geographic locations. Additionally, geometric information has not been treated in this work; however, similar concepts of spatial reasoning techniques can be applied to Web documents that include geometric information represented by, for example, the Geographic Markup Language GML [32].

Acknowledgments

This work has been funded by Millennium Nucleus Center for Web Research, grant P01-029-F, Mideplan, Chile.

References

- [1] McCurley, K.: *Geospatial Mapping and Navigation of the Web* (eds.): 10th International World Wide Web Conference on the World Wide Web, Hong Kong: ACM Press, NY (2001)
- [2] Voorhees, E.: Using WordNet for Text Retrieval. in Fellbaum, C. (eds.): *WordNet: An Electronic Lexical Database*, The MIT Press, Cambridge, MA, (1998) 285-303
- [3] Jiang, J. and Conrath, D.: Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy(eds.): *International Conference on Computational Linguistics (ROCLING X)*, Taiwan (1997)
- [4] Guarino, N. and Giarretta, P.: *Ontologies and Knowledge Bases: Towards a Terminological Clarification*. in Mars, N. (eds.): *Toward Very Large Knowledge Base: Knowledge Building and Knowledge Sharing*, IOS Press, Amsterdam, The Netherlands, (1995) 25-32
- [5] Goñi, A., Illarramendi, A., Mena, E., and Blanco, J.: An Optimal Cache for a Federated Database System. *Journal of Intelligent Information Systems*. 9:(2) (1997) 125-155
- [6] Laurini, R. and Thompson, D.: *Fundamentals of Spatial Information Systems*. Academic Press, San Diego, CA (1992)
- [7] Bruns, T. and Egenhofer, M.: Similarity of Spatial Scenes. in Kraak, M. and Molenaar, M. (eds.): *Seventh International Symposium on Spatial Data Handling (SDH '96)*, Delft, The Netherlands, (1996) 4A.31-42

- [8] Randell, D., Cui, Z., and Cohn, A.: A Spatial Logic Based on Regions and Connection. in Nebel, B., Rich, C., and Swarthout, W. (eds.): *Principles of Knowledge Representation and Reasoning*, KR '92, St. Charles, IL: Morgan Kaufmann, Cambridge, MA, (1992) 165-176
- [9] Hernández, D.: *Qualitative Representation of Spatial Knowledge*. Lecture Notes in Artificial Intelligence (Sub series of Lecture Notes in Computer Science), ed. Carbonell, J. and Siekmann, J. Vol. 804. Springer-Verlag, Berlin (1994)
- [10] Priss, U. and Old, J.: *Information Access Through Conceptual Structures and GIS*(eds.): *Information Access in the Global Information Economy*, 61st Annual Meeting of ASIS, (1998)
- [11] Jones, C., Taylor, C., Tudhope, D., and Baynon-Davies, P.: *Conceptual, Spatial, and Temporal Referencing of Multimedia Objects: Advances in GIS Research II*. in Kraak, M. and Molenaar, M. (eds.): 7th Symposium on Spatial Data Handling, Delf, The Netherlands, (1996) 12-26
- [12] Alani, H., Jones, C., and Tudhope, D.: *Associative and Spatial Relationships in Thesaurus-Based Retrieval*. in Bordinha, J. and Baker, T. (eds.): *Fifth European Conference on Advances in Digital Libraries*, Lecture Notes in Computer Science, Springer-Verlag, Lisbon, Portugal, (2000) 45-58
- [13] Hong, J.-H., Egenhofer, M., and Frank, A.: *On the Robustness of Qualitative Distance- and Direction-Reasoning*. in Peuquet, D. (eds.): *Autocarto 12*, Charlotte, NC, (1995) 301-310
- [14] Egenhofer, M., *Spatial Query Languages*. 1989, University of Maine, Orono, ME.
- [15] Clementini, E., Sharma, J., and Egenhofer, M.: *Modelling Topological Spatial Relations Strategies for Query Languages*. *Computing and Graphics*. 18:(6) (1994) 815-822
- [16] Cohn, A.: *Qualitative Spatial Representation and Reasoning Techniques*. in Brewka, G., Habel, C., and Nebel, B. (eds.): *Proceedings of KI'97*. Lecture Notes in Artificial Intelligence 1303, Springer-Verlag, Berlin, Germany, (1997) 1-30
- [17] Freksa, C.: *Qualitative Spatial reasoning*. in Frank, A. and Mark, D. (eds.): *Cognitive and Linguistics Aspects of Geographic Space*, Kluwer Academic Publishers, Dordrecht, (1991) 361-372
- [18] Egenhofer, M.: *Deriving the Composition of Binary Topological Relations*. *Journal of Visual languages and Computing*. 5:(2) (1994) 133-149
- [19] Rodríguez, A. and Egenhofer, M.: *Comparison of Inferences about Containers and Surfaces in Small-Scale and Large-Scale Spaces*. *Journal of Visual Languages and Computing*. 6:(6) (2000) 639-662
- [20] Papadias, D. and Sellis, T.: *Qualitative Representation of Spatial Knowledge*. *Very large Data Bases Journal*. 3:(4) (1994) 479-516
- [21] Frank, A.: *Qualitative Spatial Reasoning about Cardinal Directions*. in Mark, D. and White, D. (eds.): *Autocarto 10*, Baltimore, MD, (1991) 148-167
- [22] Hong, J., *Qualitative Distance and Direction Reasoning in Geographic Space*. 1994, Department of Spatial Information Science and Engineering, University of Maine, Orono, ME.
- [23] Sharma, J. and Flewelling, D.M.: *Inferences From Combined Knowledge about Topology and Directions*. in Egenhofer, M. and Herring, J. (eds.): *Advances in Spatial Databases 4th International Symposium, SSD'95*, Springer-Verlag, Berlin, Portland, ME, (1995) 279-291
- [24] Sharma, J., *Integrated Spatial Reasoning in Geographic Information Systems: Combining Topology and Direction*. 1996, University of Maine, Orono, Maine.
- [25] Vieu, L.: *Spatial Representation and Reasoning in AI*. in Stock, O. (eds.): *Spatial and Temporal Reasoning*, Kluwer Academic Publishers, Dordrecht, The Netherlands, (1997) 5-41
- [26] Florence, J., *Prediction Frequency of Topological Relations in Geographic Datasets*. In Department Spatial Information Science and Engineering. 1997, University of Maine: Orono, ME
- [27] Hernández, D., Clementini, E., and Felice, P.D.: *Qualitative Distance*. in Frank, A.U. and Kuhn, W. (eds.): *Spatial Information Theory — A Theoretical Basis for Geographic Information Systems*, International Conference COSIT'95, Verlag-Springer, Semmering, Austria, (1995) 45-58
- [28] Godoy, F. and Rodríguez, A.: *A Quantitative Description of Spatial Configurations*(eds.): *Spatial Data Handling*, Ottawa, Canada: Springer-Verlag (2002)
- [29] Guttman, A.: *R-Trees: A Dynamic Index Structure for Spatial Searching*(eds.): *ACM SIGMOD Int. Conf. on Management of Data*, (1984)
- [30] Bray, T., Paoli, J., Sperberg-McQueen, C.M., and Maler, E., *Extensible Markup Language (XML) 1.0*. 2000, World Wide Web Consortium W3C
- [31] Pepper, S. and Moore, G., *XML Topic Maps (XTM) 1.0*. 2001, TopicMaps.Org Authoring Group
- [32] OpenGIS, *GML (Geographic Markup Language)*. 2002, OpenGIS

Building Yearbooks with RDF

Ernesto Krsulovic Morales, Claudio Gutiérrez

Center for Web Research

Dept. of Computer Science, Universidad de Chile

Blanco Encalada 2120, Santiago, Chile

E-mail: {cgutierr, ekrsulov}@dcc.uchile.cl

Abstract. We present a simple application of semantic integration using the RDF model of metadata, namely, the construction and maintenance of a yearbook. It can be built and used by organizations which already have their information on the Web and require to keep yearbooks to service advanced searching facilities. Unlike traditional approaches, ours ensures wide interoperability, extensibility and historical recording by using RDF and a decentralized approach.

Keywords: Semantic Integration, RDF, Semantica Web, Yearbook

1 Introduction

Yearbooks A yearbook is a periodic document issuing information (reports, statistics, etc.) about a particular subject, and is becoming a common practice to publish them in digital format on the Web. Examples are annual reports published by companies referring to departments, projects, financial information (e.g. search site:cl memoria anual in Google for examples from Chile), or association yearbooks, which contain information about the structure of each member organization associated along with activities it develops (e.g. search ' 'association yearbook' ', in Google for representative examples.) From a semantic point of view, a classic yearbook is directed to a restricted set of users and usually the process of consulting it is mainly a human-oriented task.

Building yearbooks is a typical task in many organizations which, although straightforward at first sight, becomes complicated when the organization has strong hierarchies or several horizontal components. The classic approach to building these information centers is to delegate the task to one of the associated members, which becomes in charge of designing or updating the schema of the data, discussing it with other members, collecting the information among the different components of the organization, and populating the database. *Designing* the schema for a yearbook is a typical problem in the area of integration of information. The weakness of these traditional methodologies, though, is that they are not designed to achieve semantic persistence and extensibility, e.g. make the yearbook usable for people and applications we do not have in mind today [1]. *Collecting* the information to build a yearbook is a task not easy to automatize: each component must, on the one hand, transform the information kept to the common schema chosen for the yearbook, and on the other, create the information needed which is not kept explicitly. If the organization is highly hierarchical, the above task is usually simplified because the organization of the information is predefined for all the organization from the top.

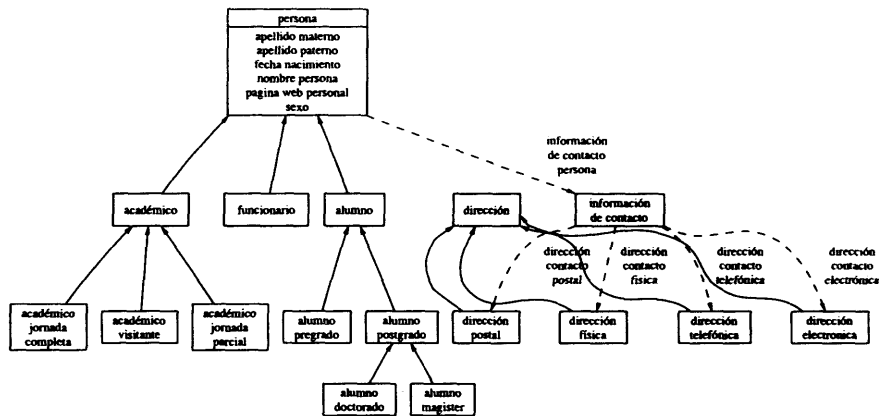


Figure 1: Fragment of the ontology corresponding to people in a university Department.

One approach to overcome the above difficulties is to use semantic integration techniques to design the yearbook, very much like in organizational memory [2], and semi-automatize the process of collecting the information via techniques developed in the framework of the Semantic Web, that is to publish periodically metadata according to an established ontology. Let us describe these frameworks.

Semantic Integration and Decentralization We claim that the construction of a yearbook of organizations on the Web is better implemented by using the framework of semantic integration [3], [4], [5] and a distributed approach [6].

To get its full potential, the information on the Web needs to be interoperable not only at a syntactic or structural level: the semantic layer is an essential one. Although this idea is at the core of the future Semantic Web [7], it is not easy to implement today. What are the tools we have at hand? It is important to note that XML was designed to provide interoperability at structure and document level. XML helps to give semantics to documents via tags with “meaning”; although this is a great advance over HTML, this “meaning” is still very much directed to humans (i.e. difficult or impossible to understand by software agents) and expresses mainly the structure of documents. For a particular domain, XML can be a good choice to express meaning and define common vocabularies trough XML Schema [8]. But when it comes to interoperability of applications across several independent domains, the limitations of XML becomes apparent [3], [9]. The solution proposed by the W3C is a framework for semantic interoperability, namely RDF, described below.

There are several schemas of cataloguing on the Web, probably the most popular are indexes like *Yahoo*. Problems with such indexing schemas are the tradeoff between the number of sites indexed and the precision and type of the description (at document level), and the fact that creation and maintenance of metadata is kept centralized. Although indexes such as DMOZ [10] have alleviated cataloguing labor by using voluntary editors by categories, still persist the problems associated to centralized maintenance and storing, like the size of the data, reliability, general descriptions at document level.

A solution to the problem of centralization is to provide the necessary infrastructure to

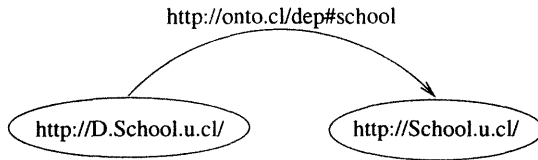


Figure 2: Example of an RDF graph

allow each site to keep its own metadata. At metalanguage level the solution recommended by the W3C is RDF [11]. The solution to the problem of quality of descriptions is the creation of shared vocabularies by topics or domains and a schema to exchange those vocabularies.

The RDF framework *RDF (Resource Description Framework)* is the metadata model and framework proposed by the W3C, and is rapidly gaining popularity as a de facto standard [11]. Using RDF instead of plain XML (not to mention HTML) offers several advantages: (1) a better model from a semantic point of view; (2) full semantic extensibility; (3) facilities to handle distributed data. All these features are very desirable when building a distributed and extensible yearbook. On the Web, data are documents, multimedia objects and links; these data are all called resources. RDF metadata is simply a set of statements about resources using predicates from shared vocabularies. The RDF model strongly supports extensibility. A relational or XML database has a very static schema and it is very difficult to make any significant changes without impacting existing code. Compared to XML, the RDF model is dynamic and it is easy to add new information to an existing description. Moreover, when keeping metadata with RDF it is possible to scale solutions by integration of previously disjoint organizations.

RDF offers a better semantic model than existing alternatives due to the simplicity of its model (a labeled graph where order is not relevant) and its flexibility to describe data (a simple subject-predicate-object model) [8]. In RDF, resources are identified by a Uniform Resource Identifier (URI), an identification schema which generalizes URLs. In an RDF statement predicates as well as subjects are URIs and the object is a URI or a literal. So for example, it is possible to register the statement “Department D belongs to School S” with the following triple:

```

<http://onto.cl/dep#>
<http://D.School.u.cl/>
<http://School.u.cl/>

```

where the first is the predicate’s URI indicating to what School belongs the given department, the second identifies the department, and the third URI identifies the School. The graph corresponding to this statement is shown in Figure 2.

To describe information on the Web with RDF, we need a common vocabulary, i.e. an *ontology*. One such example is Dublin Core, a minimal vocabulary to describe title, author, copyrights, etc. of a document [12]. For our purposes we need more specific ontologies. RDF allows the specification of such ontologies through RDF Schema (RDFS), which is modelled by classes, sub-classes, hierarchies, relations and properties of classes [13]. For example, in our prototype Yearbook which models university Departments, we need to model people

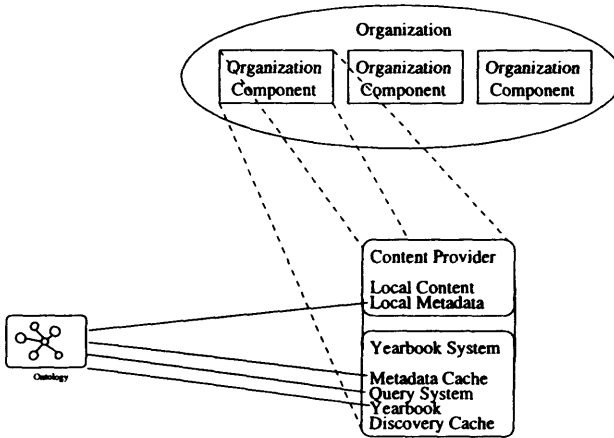


Figure 3: Yearbook system and sites of the organization components.

who belong to a Department and their contact addresses, obtaining a hierarchy of classes, attributes of the class *persona* and the relations among classes as shown in Figure 1.

Handling distributed data with RDF is simple because, as described above, an RDF specification is a collection of atomic sentences over a shared common vocabulary built by different parties. Although storing atomic sentences over diverse vocabularies currently can be a drawback when the size of the data is huge, this is an issue being extensively investigated and resembles the problem of storing efficiently relational data in the early seventies or XML data at the end of the nineties.

Yearbooks versus Catalogs: t-books To finalize the introductory discussion, let us make some remarks about the difference between a yearbook and a catalog in our context and some tips learned about temporality of metadata.

A catalog is a list of items, usually ordered, whose goal is describing data in a succinct way, e.g. exhibition catalogs, library catalogs, shopping catalogs. The main implicit characteristic of a classical catalog is the hidden assumption about the invariability (atemporality) of the data it describes. (This is more obvious in the shopping catalogs, where price, season, etc. are key parameters). On the Web, catalogs turned exactly into its opposite: although they still do not have a temporal parameter, its information –in the form of links to web sites– is highly variable. In fact, any attempt to describe data on the Web faces the problem of (extremely and unpredictable) variability of data. For many temporal applications this is not important: the catalog of *amazon.com* changes daily, and probably this is a feature more than a problem. But for other kind of information this is a crucial point. Consider for example a catalog of graduate students or financial statistics.

Yearbooks can be considered essentially catalogs with a fixed time-stamp (is better to call them *t-books*) and secondly, less important for our discussion, that contain aggregate data (statistics). The periodicity of the yearbook usually indicates an actual change of data: new year, new semester, new month, new balance, etc. But several applications need a different timing: bi- or tri-annual for university courses, montly for accounting, daily for newspapers.

We think the concept of yearbooks is a good “discrete approximation” to an online catalog on the Web, useful for several applications, and has the additional advantage that, caching mediating, can be used as historical record, hence providing temporal information usually lost in current “on-line catalogs”. From the point of view of the Semantic Web, it has additional advantages: (1) allows historical recording, hence providing temporal information; (2) simplifies marking (with metadata) by restricting it to a smaller group of people, hence more specialized. From a different point of view, a t-book is a compromise between a centralized system like DMOZ and completely distributed system like Edutella [14].

In what follows, we will use “yearbook” to mean a t-book for some $t > 0$.

2 A decentralized Yearbook System

2.1 Architecture

Each component of the organization can offer two services: *Content Provider*, that is, provides the information contained in its Web site, and *Yearbook System*, an optional service for the components, although it must be offered at least by one of them (see Figure 3).

The modules of a Content Provider are:

- *Local Content*. Typically static or dynamic Web pages, documents and multimedia content. This information is already in the Web site of each component.
- *Local Metadata*. Markup of local contents in the form of RDF triples, using ontologies defined by the organization. Typically these metadata will not be in the Web site of the component, and must be generated according to the periodicity defined by the designers of the Yearbook.

The Yearbook System gives the following services to the system:

- *Metadata Cache*. Stores copies of the metadata of producers. The metadata is periodically updated according to the system schedule, and keeps historical records of previous versions.
- *Query System*. Provides query functionality to the global metadata of the system. This module is one of the building blocks of the whole system.
- *Yearbook*. Allows “navigation” among metadata and does the processing of predefined queries defined by a hierarchycal structure or a fixed taxonomy. Additionally it provides a searching-by-pattern facility at different levels.
- *Discovery Cache*. Keeps a list of known producers. This module allows new yearbook services to get into the system through one of the components already connected to the network.

2.2 A case of study

We will present a prototype yearbook for the academic Departments of Chilean universities. This domain is attractive because the components (university departments) have great interest in such a system, the organization (Council of Universities, Ministry of Education) have

shown interest in incorporating new technologies to their system. Additionally, this is a case where the organization is highly horizontal, with weak leadership, but for students, educators and researchers this group of components form a very natural unit. We started with the particular case of Computer Science Departments because they are a manageable number in Chile, all of them have Web sites, and obviously can participate more actively in the project which involves technologies familiar to them. It is interesting to remark that today we do not have in Chile any similar system, centralized or not, which performs the functionalities we have described for a yearbook. The need for such a system is out of discussion.¹

Ontology The process of design (or reuse and adaptation, if it is available) of an ontology is among the most important aspects of the project and *is the semantic glue among the modules of the proposed decentralized Yearbook*. We used Protégé [15] and followed classical methodologies [16]: building the vocabulary by filtering it from the Web pages of the Chilean Departments of Computer Science, organizing it in classes, determining attributes, relationships and properties. Then we iterated this process through each of the Web pages of the CS Departments. The design is still a preliminar one, and we believe it can be the basis for the ontology of chilean university Departments.

It is worth mentioning that the Academia has been used before as a test bed for markup projects, and university ontologies exist. Unfortunately for us, their concepts, vocabulary and emphasis are far from those used in departments in Chile. Considering this, the best solution we found was to build a local ontology and then specify, at the right level, the conceptual translations to other university ontologies. Below is a brief description of the Ontology. For details see [17].

- Vocabulary for a university Department: 5 abstract classes, 16 classes and 54 attributes to mark admission, teaching and research information, and to describe people belonging to it: students, academic and support staff.
- Vocabulary to describe contact information: 1 abstract class, 5 classes, 17 attributes. It allows to basically specify personal and institutional addresses in four different formats: electronic, physical, telephonic and regular mail.
- Taxonomy of research areas: 28 abstract classes, enough to catalog research areas of people, institutions and research centers existing in Chile (2002).

Local Content Most of the information required by the Yearbook is already in the Web site of each Department in HTML format.

Local Metadata We used Protégé as markup tool. Protégé provides forms to create instances given an ontology and has a facility to export metadata in RDF format. We remark that this tool, as several others available today, requires some training to get all its functionality, particularly the markup process. In the case of Yearbooks, this is not a big problem because usually organizations which build Yearbooks already have dedicated personnel, hence there is no extra investement or effort to orient them to learn this new process.

¹Today a person needs to navigate manually through a highly heterogeneous set of Web sites to find elementary aggregate information about these departments.

Local metadata is stored in one or few central RDF files that contain all instances of classes for all pages of each Web site of each Department.

Metadata Cache In our current implementation we do not keep cache of metadata. Our experience shows that this is a necessity to scale the system and give it a minimum of efficiency. Our plan is to store metadata in a relational database using the feature of *SQL backend support* of Inklink [18].

Query system The query system implemented provides the functionality of SquishQL [19], a query language over RDF allowing queries very much like SQL. We chose this alternative among several query languages available because it provides a base implementation easy to fit our goals and allows queries on several RDF sources at one time by just writing up different URIs in the FROM predicate. See Figure 4.

```
SELECT ?curso, ?departamento, ?univ
FROM
  http://purl.org/net/depmark/puc,
  http://purl.org/net/depmark/uch,
  http://purl.org/net/depmark/umag,
  http://purl.org/net/depmark/udec,
  http://purl.org/net/depmark/utfsm,
  http://purl.org/net/depmark/usach
WHERE
  (depmark::cursos_del_departamento ?docencia ?idcurso)
  (depmark::nombre_curso ?idcurso ?curso)
  (depmark::docencia_departamento ?iddep ?docencia)
  (depmark::nombre ?iddep ?departamento)
  (depmark::universidad_a_que_pertenece ?iddep ?univ)
USING depmark for http://purl.org/net/depmark#
```

Figure 4: A query asking the courses of all departments.

Yearbook The Yearbook system is in development and will contain a Web interface to query and navigate over the Yearbook information. One problem not completely solved in our prototype is a simple and flexible interface to present all the richness of the query language.

Discovery Cache The list of known Providers is kept in an RDF file which contains the URI where the actual local metadata of each component is.

Extensibility The plan is to extend this Yearbook to all chilean Departments. This is not an easy task, not only due to the diverse ontologies needed (for each area or type of Department), but also to the difficulty to introduce brand new technologies in other Departments besides Computer Science's.

3 Conclusions

Lessons from the prototype Building an ontology requires a deep knowledge of the domain to be modelled: something already known, but worth repeating. It was very useful in the process of building the ontology to have at hand typical examples of pages we wanted

to describe: first for creating the vocabulary and relations and second to test the “correctness and completeness” of the ontology.

One of the problems faced was the lack of tools to build the system, especially in the area of markup. This problem was already reported and studied [20], but the problem seems to be deeper. We found that there are not only problems in the process of markup or maintenance of it, but also at the level of deciding at each stage if it is worth continuing with such process.

The existence of different roles of the contents show the existence of *spaces of markup*, that is, different views of the same contents which must be marked with different ontologies and kept in different spaces. This is an important remark for developers of markup tools, which should consider a feature to deal with this problem.

General conclusions We believe that an application of low complexity as a Yearbook allows testing principles of semantic integration on the Web without facing the problems usually found when using RDF in Web systems, particularly critical mass, markup tools and system development and maintenance [20]. Critical mass is not necessary but at the level of the organization. Due to the characteristic of a Yearbook, we can have trained people (one in each component) to do the markup process. The ontology development is basically a formalization of an informal ontology already used in practice and implicitly on documents. One of the main problems of Semantic Web applications, that of being very unfriendly, is in the Yearbook application not a problem: the system is designed to be built by few people in charge at each component. But the fruits will be harvested by all members of all components.

Summarizing, we presented an application which: Solves the creation-of-metadata bottleneck (by restricting to a particular domain of application); Extend functionality of current Yearbooks by making them interoperable from a semantic point of view; Suggests an approach to build catalogs with a discrete (periodic) temporal parameter; Allows a simple way of implementing massively historical memory. Finally, last but not least, will give the Chilean academic community a reliable, rich and flexible way of querying the core organizations of our Universities: the Departments.

Acknowledgements Funded by Millenium Nucleous Center for Web Research, Grant P01-029-F, Mideplan, Chile.

References

- [1] Henry Kim. Predicting how ontologies for the semantic web will evolve. *Communications of the ACM*, 45(2):48–54, February 2002.
- [2] Fabien Gandon, Laurent Berthelot, and Rose Dieng-Kuntz. A multi-agent platform for a corporate semantic web. In C. Castelfranchi and W. Johnson, editors, *AAMAS 2002, 6th International Conference on Autonomous Agents, 5th International Conference on Multi-Agents Systems, 9th International Workshop on Agent Theories Architectures and Languages*, pages 1025–1032, July 2002.
- [3] Stuart Madnick. The misguided silver bullet: What xml will and will not do to help information integration. In *Information Integration and Web-based Applications & Services (IIWAS2001)*, September 2001.
- [4] A. Sheth. Changing focus on interoperability in information systems: From system, syntax, structure to semantics. In M. F. Goodchild, M. J. Egenhofer, R. Fegeas, and C.A. Kottman, editors, *Interoperating Geographic Information Systems*, pages 5–3, Kluwer, 1998. Academic Publishers.
- [5] H. Wache, V. ogele, T. Visser, U. Stuckenschmidt, H. Schuster, G. Neumann, and H. ubner. Ontology-based integration of information - a survey of existing approaches, 2001.

- [6] D. Dornfest and D. Brickley. The power of metadata. <http://www.openp2p.com/pub/a/p2p/2001/01/18/metadata.html>, 2001.
- [7] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, May 2001.
- [8] Varun Ratnakar and Yolanda Gil. A comparison of (semantic) markup languages. *Proceedings of the 15th International FLAIRS Conference, Special Track on Semantic Web*, May 2002.
- [9] Peter Patel-Schneider and Jérôme Siméon. The yin/yang web: Xml syntax and rdf semantics. In *The Eleventh International World Wide Web Conference, WWW2002*, 2002.
- [10] Open directory project. <http://dmoz.org/>.
- [11] Ora Lassila and Ralph Swick. Resource description framework (rdf) model and syntax specification. <http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/>, 1999.
- [12] Dublin core metadata initiative (dcmi). <http://purl.oclc.org/dc/>.
- [13] Dan Brickley and R.V. Guha. Rdf vocabulary description language 1.0: Rdf schema. <http://www.w3.org/TR/2002/WD-rdf-schema-20020430/>, 2002.
- [14] Wolfgang Nejdl, Boris Wolf, Changtao QuLearning, Stefan Decker, Michael Sintek, Mikael Nilsson, Matthias Palmér, and Tore Risch. Edutella: A p2p networking infrastructure based on rdf. In *The Eleventh International World Wide Web Conference, WWW2002*, May 2002.
- [15] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen. Creating semantic web contents with Protégé-2000. *IEEE Intelligent Systems Journal*, 16(2):60–71, 2001.
- [16] Natalya Fridman Noy and Deborah L. McGuinness. Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, Stanford Knowledge Systems Laboratory, March 2001.
- [17] Ernesto Krsulovic and Claudio Gutierrez. Depmark, marcado con metadatos de departamentos universitarios chilenos. <http://purl.org/net/depmark/>.
- [18] Libby Miller. Inkling: Rdf query using squishql. <http://swordfish.rdfweb.org/rdfquery/>.
- [19] Libby Miller, Andy Seaborne, and Alberto Reggiori. Three implementations of squishql, a simple rdf query language. In *1st International Semantic Web Conference (ISWC2002)*, Sardinia, Italy., 2002.
- [20] Stefan austein and Jörg Pleumann. Easing participation in the semantic web. In *Workshop at WWW2002, International Workshop on the Semantic Web*, May 2002.

A non-Deterministic versus Deterministic Algorithm for Searching Spatial Configurations

Mary Carmen Jarur and M. Andrea Rodríguez
Department of Information Engineering and Computer Science
University of Concepción
Edmundo Larenas 215, Concepción, Chile.
{mjarur, andrea}@udec.cl

Abstract. A deterministic approach to searching spatial configurations can be a computationally demanding task, since it implies to permute combinations of possible objects stored in a database in order to satisfy particular spatial constraints. In addition, while deterministic approaches may find the best solution, they can also miss possible solutions due to *local optimum* effects. In this paper, we present an approach to searching spatial configurations that explores the characteristics of genetic algorithms to find solutions within a time framing. The novelty of our approach lies in the combination of genetic algorithms with a *heuristic operator* and an *indexing schema* for handling binary spatial constraints. Experimental results compare a genetic versus a deterministic algorithm and show the convenience of using a genetic algorithm depending on the type and complexity of a user query.

1 Introduction

The growing amount of multimedia data available in the WWW, as well as the need for automating searching processes, have made the content-based retrieval of spatial configurations a challenging area of investigation. In this paper, we refer to spatial information as any information with a spatial dimension, that is, information that includes geometry or the spatial arrangement of its components within a physical space. Using the object-oriented paradigm, a spatial configuration is composed of a set of spatial objects with geometric characteristics, such as size and shape, and a set of spatial relations, such as topology, orientation and distance, between these objects. In this context, the set of geometric characteristics and the set of spatial relations represent the information content of the configuration and define the constraints upon which a content-based retrieval is performed.

The retrieval of spatial configurations means to search *object instances* in a data collection whose spatial interrelations similar to the relations between *object variables* of a *user query*. Under this perspective, we do not impose a fix number of objects in a user query and we avoid the treatment of predefined groups of objects in the data collection as predefined configurations. The latter case corresponds, for example, to a collection of raster images, where each image is considered a configuration and user queries are of equal or similar size to the images [1-3]. The type of query we addressed is seen, therefore, as a type of Constraint Satisfaction Problem CSP [4], where constraints are topological relations between pairs of objects.

A number of studies have defined searching strategies of spatial configurations [1, 5-10]. These studies are characterized for using deterministic algorithms, that is, algorithms that always give the same answer to similar inputs and whose response times depend on the complexity of the query and data collection. This paper extends our previous work on similarity-based searching of spatial configurations based on deterministic algorithms [14]. It explores the use of genetic algorithms with heuristics applied to a CSP using a content measure of spatial configurations. Genetic algorithms are non-deterministic algorithms that always find solutions by considering, at each cycle of an evolutionary process, randomly selected objects taken from the whole space of possible solutions. In this way, they are not misled by *local optimums* and they may be limited to a time framing. In our case, a *local optimum* occurs when there exist good partial results that reduce the search domain of instances and produce a bad global result. In this study, we aim at finding good global results.

Few previous studies have addressed the problem of searching spatial information with genetic algorithms [11-13]. Focusing just on spatial configurations Papadias *et al.* [9, 11] have designed a genetic algorithm for answering structural queries, that is, queries based on spatial relations. Unlike their work that uses classic genetic operators, our work defines a new heuristic operator that incorporates a content measure of spatial configurations with an indexing schema for handling spatial constraints.

The organization of the paper is as follows. Section 2 describes the framework used to compare and organize spatial relations. Section 3 presents general characteristics of genetic algorithms. Section 4 introduces the algorithm implemented. Experiments using a randomly created data set are presented as a proof of concepts in Section 5. Conclusions and future research directions are presented in Section 6.

2 Framework for Searching Spatial Configurations

The problem of searching spatial configurations is expressed by a user query with a set of n object variables $Q = \{v_1, \dots, v_n\}$, where objects belong to equal or different semantic classes. In the approach of this work, the query constraints are the objects' classes and the objects' topological interrelations.

In our previous work [14-15], we characterized the relations between objects by a content-measure of objects' MBRs. Unlike other studies [3,7,16-17], this content measure is a continuous function that quantitatively distinguishes topological relations. The content measure considers three primitives over objects' MBRs: areas of individual MBRs and areas of intersection of pairs of MBRs, diagonals of MBRs, and minimum internal and external distances between boundaries of MBRs (i.e., $d_i(\delta A, \delta B)$ and $d_e(\delta A, \delta B)$, respectively) (Figure 1) (Equation 1).

$$\begin{aligned}
 F_m(A, B) &= \frac{\text{area}(A) - 2\text{area}(A \cap B)}{\text{area}(A)} + \frac{\text{distance}(\partial A, \partial B)}{\text{diagonal}(A)}, \\
 F_m(B, A) &= \frac{\text{area}(B) - 2\text{area}(A \cap B)}{\text{area}(B)} + \frac{\text{distance}(\partial A, \partial B)}{\text{diagonal}(B)} \text{ where} \\
 \text{distance}(\partial A, \partial B) &= \begin{cases} d_e(\partial A, \partial B) & \text{if } A \cap B \neq \emptyset \\ -d_i(\partial A, \partial B) & \text{if } A \cap B = \emptyset \end{cases}
 \end{aligned} \tag{Eq.1}$$

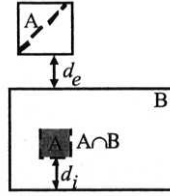


Fig. 1. Basic MBR's descriptors

The content measure distinguishes the eight topological relations between regions [18] with their variations depending on relative sizes and distances between MBRs [19]. In this context, searching spatial configurations implies to find solutions that contain n instances of objects in the data collection $S = \{u_1, \dots, u_n\}$. Given that the relation between a pair of query objects v_i and v_j is described by a pair of values $F_m(v_i, v_j)$ and $F_m(v_j, v_i)$, the search process looks for pairs of instances u_i and u_k whose values $F_m(u_i, u_j)$ and $F_m(u_i, u_j)$ are similar to $F_m(v_i, v_j)$ and $F_m(v_j, v_i)$.

The global similarity of a solution with respect to a query (i.e., the degree of fitness) is then determined by the following sum of distances

$$D(Q, S) = \sum_{v_i, v_j \in Q, u_i, u_j \in S} \sqrt{(F_m(v_i, v_j) - F_m(u_i, u_j))^2 + (F_m(v_j, v_i) - F_m(u_j, u_i))^2} \quad (\text{Eq. 2})$$

between content measures calculated for each pair of variables in the query and the corresponding pair of instances in the data collection. According to Equation 2, the maximum degree of similarity is obtained if D is zero.

We use a domain of searching composed of a large number of objects $N \gg n$, where N is the number of instances in the data collection and n is the number of object variables in the user query. In this domain, searching implies to perform all n permutation over the N objects, that is, it scales exponentially in the size of the query ($O(N^n)$). In order to improve performance, previous studies have used a forward checking approach with heuristics over traditional spatial indexing schema (R*Tree) [7-9, 20]. We have shown recently that, by indexing relations rather than spatial location, we could reduce the number of comparisons and, therefore, the complexity of the searching algorithms. Consequently, in our work we used an indexing schema over relations, that is, we indexed points in the 2D space conformed by the pairs of values of spatial relations [14].

3 Alternative Genetic Approaches to CSP

Genetic algorithms (GA) are search methods that are based on concepts of Darwinian evolution. They use a performance *criterion* for evaluation and an initial *population* of possible solutions to the search for a global optimum. The manipulation of the *population* is given by a set of *genetic operators* that works on the population's *chromosomes* or *individuals*, which are composed of a set of *alleles*. In each *generation* or *cycle* of the GA, the new solutions arise from the application of genetic operators. The fitness function, quantified by the objective function, represents the individuals' ability for surviving. Basic *genetic operators* are:

- *Selection*: It takes the more fit individuals for creating a new population. These selected individuals become the domain of possible parents for the *crossover* operator.

- **Crossover:** It creates new individuals by combining parts from several (two or more) individuals ($c: S \times \dots \times S \rightarrow S$).
- **Mutation:** It modifies individuals by a small change ($m: S \rightarrow S$).

Although a large number of studies use genetic algorithms in optimization applications [21], CSP problems involve constraints that are not naturally treated with traditional genetic operators [22]. Indeed, operators such as mutation, selection, and crossover are blind to the constraints. Hence, whether or not parents satisfy the query constraints does not imply that their offspring will do. The main issue in CSP problems is, therefore, how to incorporate constraints in an evolution cycle.

Handling binary constraints with genetic algorithms have several alternatives. An indirect strategy considers fitness functions that involve values associated with constraint satisfaction. Such type of strategy is suitable, since it reduces the CSP problem to a simple optimization problem and incorporates user preferences by adding weights of relevance in the fitness function. This strategy, however, hides the constraints within a global fitness function without ensuring the satisfaction of any constraints. A direct strategy, on the other hand, adapts the traditional genetic operators by including the constraints. Some adaptations to classic operators may be: *elimination of unfeasible candidates*, *preservation of feasible candidates*, *repairing of unfeasible candidates*, and *creation of new domain-depending operators* [22].

Our work combines both direct and indirect strategies by using a global fitness that depends on single constraint satisfactions (Equation 2) and by defining a new genetic operator that considers binary constraints of spatial relations, which is described in the following Section.

4 A Genetic Algorithm for Searching Spatial Configurations

Following the ideas of Eiben *et al.* [23-24], we defined a GA that incorporates CSP heuristics into genetic operators, an *Asexual Heuristic Algorithm* (AHA). The defined GA includes operators of the classic GA [25] (Figure 2), but it replaces the *crossover* operator by an *asexual heuristic* operator (AH). This operator applies heuristics related to the satisfaction of constraints as well as it uses an index that organizes the domain of instances in the data collection based on these instances' spatial interrelations. Thus, as the GA searches for pairs of instances that satisfy a constraint, the index retrieves just the reduced set of instances with this condition.

```

Procedure Genetic_AHA
begin
  call Parameter_Input
  call Initial_Population
  call Population_Evaluation
  gen = 0;
  //Beginning of evolution cycle
  while (gen < Maximum_Generation) do
    call Selection
    call Asexual_Heuristic_Op
    call Substitution
    call Mutation
    call Population_Evaluation
    gen++;
  enddo
end Genetic_AHA

```

Fig. 2. General structure of Algorithm AHA

The algorithm uses a stop criterion based on the number of generations rather than a threshold of the fitness function. The number of generations limits the time of response by the systems, which ensures that even if the system does not find a good solution, the search will stop.

Components of the AHA are a query, a population, a set of parameters, a set of operators, and a fitness function. A query is defined as a variable individual, that is, a set of alleles that can take different values of instances from the data collection. The

population is composed of individuals, which are also composed of alleles. In spatial terms, a population is a set of spatial configurations. An individual is a spatial configuration and an allele is a spatial object. The CSP researches generally define the fitness function as the number of violated or satisfied constraints [21]. In our case, in contrast, we calculate a global similarity (Equation 2) for each individual respect to the query, and impute this value like the individual's fitness.

Genetic operators are methods that are applied to the population and, subsequently, to individuals and alleles. These operators are:

- *Initial Population.* The initial population is generated randomly, searching the whole domain of instances organized by their semantic classification.
- *Selection.* Selection uses a lineal ranking [19], that is, it assigns to each individual a new fitness based on the ranking position of this individual within the whole population and, it selects those individuals whose new fitness is the superior minimum value with respect to a randomly determined value.
- *Asexual Heuristic Operator (AH).* For each selected individual, it sorts the individual's alleles based on the number of constraints violated. Then, it takes the pair of alleles with the largest number of violations (i.e., the two worst alleles) and, using the index schema, searches for instances in the data collection that satisfy the constraints where these two alleles participate. Within the set of pairs of instances retrieved, it randomly selects a pair of instances and it exchanges them by the two worst alleles (Figure 3).

Procedure AH

```

for each  $I \in$  selected individuals do
    call rank_alleles // ranks alleles based on their contribution to constraint satisfaction
     $a_1 \leftarrow$  first_worst // gets the first worst allele
     $a_2 \leftarrow$  next_worst // gets the second worst allele
     $c \leftarrow$  find_constraint( $a_1, a_2$ ) // finds constraints where both  $a_1$  and  $a_2$  participate
    while (( $c \neq$  nil) and ( $ss \leftarrow$  search_index( $c$ ) = nil)) do
        // finds pairs of instances that satisfy the constraint c
         $a_1 \leftarrow a_2$ ;  $a_2 \leftarrow$  next_worst
         $c \leftarrow$  find_constraint( $a_1, a_2$ )
    enddo
    if ( $ss \neq$  nil) then
         $s \leftarrow ss[random]$  // randomly selects one pair of instances from ss
        exchange ( $a_1, a_2, s$ ) // exchanges  $a_1$  and  $a_2$  by instances in s.
    endif
enddo
end AH

```

Fig. 3. Algorithm of the asexual heuristic Algorithm AH

- *Substitution.* The strategy of substitution is elitist; that is, the individuals after the asexual heuristic operator replace the worst individuals in the population.
- *Mutation.* This operator selects, based on a probability applied randomly, a group of individuals from the whole population. For each of these selected individuals, it then selects randomly an allele that is substituted by a randomly selected instance of the data collection.

5 Experimental Results

The experiments use a data collection of 18,000 random objects of 6 different semantic classes. In this collection, *disjoint* is the relation with the highest frequency of occurrences (90%), followed by the *overlap* (7%) and *contain* relations (3%) (Figure 4). The frequency distribution of relations is due to the fact that the database is composed of objects' MBRs that are homogenously distributed over a large space. In addition, two real objects that *meet* most likely results in their MBRs *overlapping* rather than *meeting* so, the number of *meet* relations is small.

An R-Tree [26] structure was used as an index of content measures for combinations of object classes, giving rise to 21 different trees. These 21 different trees result from the symmetric combination of 6 classes plus the combination of the 6 classes with themselves, that is, $6 \times 5/2 + 6 = 21$. Nodes in each of these trees contain between 250 and 500 elements, given a maximum depth of 3 for each tree. The total number of relations indexed was 150,000, as the result of the elimination of relations whose content measures in both directions were larger than 4.0 (i.e., $d = 4.0$).

The first part of the experiments tried to find a good setting of parameters for the AHA. In particular, we tried different numbers of individuals in a population (i.e., from 50 to 300) and different numbers of generations (from 100 to 500). The probability of mutation and the percentage of individual selection were set to 1% and 70%, respectively, based on the results of previous studies [9,11]. For each of the settings, we run the AHA 20 times and determined an average of fitness for the best solution. The results are presented in Figure 5. In terms of performance, we determined the number of distance calculations and we noted that increasing the number of generations to the double of its size doubles the execution time. The same situation occurs when the number of individuals is doubled. Based on these results, we set for the following experiments the number of individuals and the number of generations to 50 and 200, respectively.

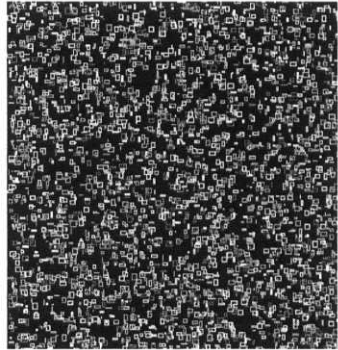


Fig. 4. Portion of the data collection

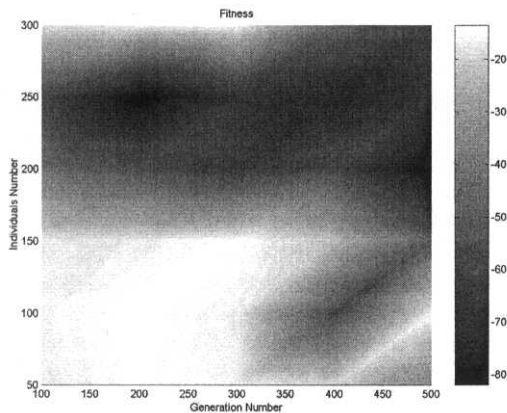


Fig. 5. Variations of fitness depending on changes in numbers of individuals and generations for the same query.

In order to compare our genetic algorithm with a deterministic one, we used our previous work on similarity-based searching of spatial configurations. This previous work defined a deterministic algorithm that uses a forward checking strategy for a constraint satisfaction problem (full description is found in [14]). For both algorithms we run a set of 20 queries, 10 queries created by selecting objects existing in the data collection and 10 queries created randomly. In the case of the genetic algorithm, we processed each query 5 times, and we took the best results. As it was done in [14], before running the search process, we preprocessed each query in order to create a sorted list of relevant constraints. The relevance criterion considers that *non-disjoint* relations are more relevant, because they indicate physical connection between objects [27].

In terms of performance, we compared algorithms based on the number of distance calculations, number of visited nodes in the index structure, and CPU time. The distance calculation and the search of pairs of instances in the data collection are the most expensive or most recurrent operations in the algorithms. These results are shown in Figure 6, where equal values for both algorithms appear as points on the 45° line, and larger values of the AHA appear as points above the 45° line. These results show that the time of response varies across different queries. While in the deterministic algorithm the number of visited node in the index structure is always less than the number of nodes in the AHA, the number of distance calculations in the former can be considered larger than the calculations in the latter. Indeed, the worst cases in response time were obtained with the deterministic algorithm. A reason is that in the case of the deterministic approach, the distance calculation is performed each time a new pair of instances that satisfies a constraint is found. Although the search of these instances is done only once during the whole process, many instances may result for each constraint. In the case of the AHA, on the other hand, in each generation and for each individual of the population, the algorithm replaces the worst pair of instances for a new pair of instances in the data collection. So, constraint searching is performed in each cycle for all individuals in the population. Consequently, it is natural to expect a large number of visited nodes in the AHA. Note, however, that the CPU time that takes searching instances is less than the CPU time for a distance calculation.

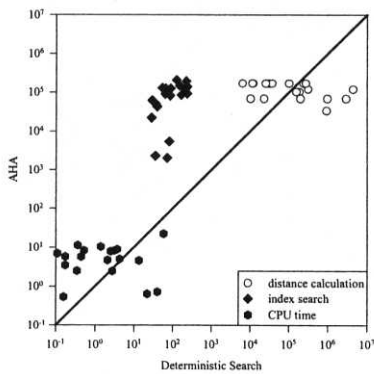


Fig. 6. Performance comparison between the AHA and the Deterministic Algorithm

For space restrictions, we just show partial results of the searching process in Figure 7. The deterministic algorithm always finds the correct solution, or best solution, for existing queries. The genetic algorithm, on the other hand, may fail in finding the best solution, but it always finds at least one. In the case of non-existing queries, the deterministic algorithm was able to find solutions in 40% of the queries.

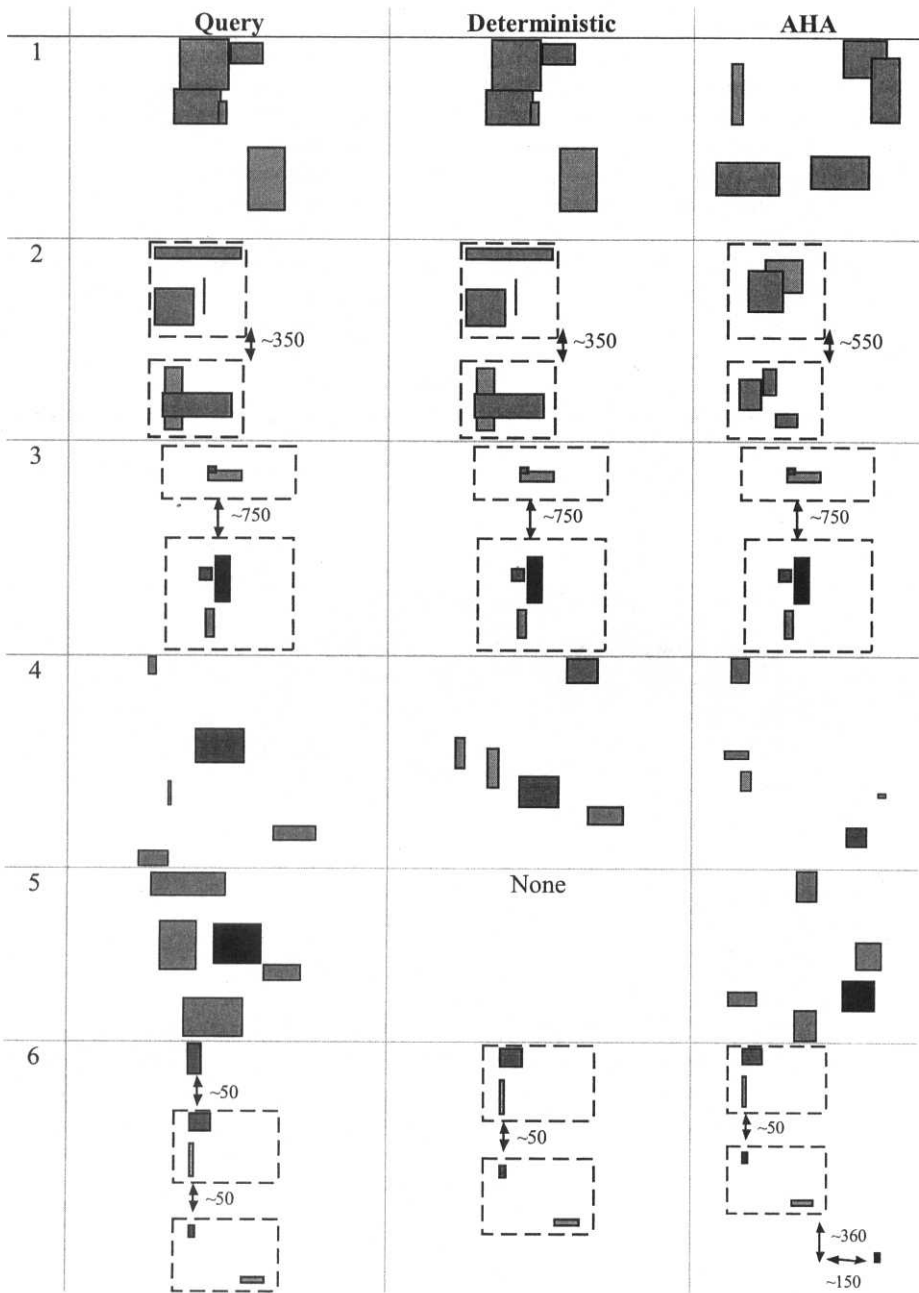


Fig. 7. Partial experimental results: (1)-(3) are existing queries in the data collection, and (4)-(6) are non-existing queries.

6 Conclusions

This work has presented a new genetic algorithm for searching spatial configuration. This algorithm is based on an *asexual heuristic operator* that replaces the classical crossover operator. This new operator was introduced in order to consider explicitly the spatial constraints between objects in a configuration. Comparing the genetic algorithm with a deterministic algorithm shows advantages and disadvantages. Advantages of the genetic algorithm are that it always finds a solution and it shows to be in the extreme cases, less time expensive than the deterministic approach. On the other hand, the deterministic approach can always find the best solution (an exact solution) when there is an exact correspondence between a set of instances in the data collection and the set of objects of the user query.

For future work, we will continue testing the algorithm for individuals with more alleles, since we expect that, in these cases, the performance of the AHA will clearly overcome the performance of the deterministic algorithm. We are also studying new genetic operators that can also treat the spatial constraints within the non-deterministic algorithm.

Acknowledgments

This work has been funded by CONICYT - Chile, under Fondecyt grant 1010897. Millenium Nucleus Center provided additional funding for Web Research, grant P01-029-F, Mideplan, Chile. We also want to thank Ricardo Contreras for his valuable comments on this work.

References

- [1] Gudivada, V. and Jain, R.: Design and Evaluation of Algorithms for Image Retrieval by Spatial Similarity. *ACM Transactions on Information Systems*. 13:(2) (1997) 115-144
- [2] Faloutsos, C., Barber, R., Flickner, M., Hafner, J., Niblack, W., Petrovic, D., and Equitz, W.: Efficient and Effective Querying by Image Content. *Journal of Intelligent Information Systems*. 3 (1994) 231-262
- [3] Petrakis, E. and Orphanoudakis, S.: Methodology for the Representation, Indexing, and Retrieval of Images by Content. *Image and Vision Computing*. 11 (1993) 505-420
- [4] Meseguer, P.: Constraint Satisfaction Problems: an Overview. *AICOM*. 2:(1) (1989) 3-17
- [5] El-Kwae, E. and Kabuka, M.: A Robust Framework for Content-Based retrieval by Spatial Similarity in Image Databases. *ACM Transactions on Information Systems*. 17:(2) (1999) 174-198
- [6] Ahmad, I. and Grosky, W.: Spatial Similarity-Based Retrievals and Image Indexing by Hierarchical Decomposition. *International Database and Engineering Applications Symposium*, (1997)
- [7] Papadias, D., Arkoumanis, D., and Karacapilidis, N.: On the Retrieval of Similar Configurations. in Poiker, T. and Chrisman, N. (eds.): 8th International Symposium on Spatial Data Handling, International Geographical Union, Vancouver, (1998) 510-521
- [8] Papadias, D., Mamoulis, N., and Delis, V.: Algorithms for Querying Spatial Structure. 24th VLDB Conference, New York, NY (1998)
- [9] Papadias, D., Mantzouroguannis, M., Kalnis, P., Mamoulis, N., and Ahmad, I.: Content-Based Retrieval using Heuristic Search. *ACM-SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, CA (1999)

- [10] Smith, J. and Chang, S.-F.: Integrated Spatial Query and Feature Image Query. *Multimedia Systems*. 7 (1999) 129-140
- [11] Papadias, D.: Hill Climbing Algorithms for Content-Based Retrieval of Similar Configurations. *ACM Conference on Information Retrieval*, Athens (2000)
- [12] Abe, K., Taketa, T., and Nunokawa, H.: An Efficient Information Retrieval Method in WWW Using Genetic Algorithms. *International Workshops n Parallel Processing*: IEEE Press (1999)
- [13] Kato, S.: An Image Retrieval Method based on a Genetic Algorithm. *International Conference on Information Networking*: IEEE Press (1998)
- [14] Rodríguez, A. and Godoy, F.: A Content-Based Approach to Searching Spatial Configurations. in Egenhofer, M and Mark, M. (eds.): *GIScience 2002, Lecture Notes en Computer Science*, Springer-Verlag, Boulder, Colorado, (2002)
- [15] Godoy, F. and Rodríguez, A.: A Quantitative Description of Spatial Configurations. *Spatial Data Handling*, Ottawa, Canada: Springer-Verlag (2002)
- [16] Bruns, T. and Egenhofer, M.: Similarity of Spatial Scenes. In Kraak, M. and Molenaar, M. (eds.): *Seventh International Symposium on Spatial Data Handling (SDH '96)*, Delft, The Netherlands, (1996) 4A.31-42
- [17] Berretti, S., Bimbo, A.D., and Vicario, E.: The Computational Aspect of Retrieval by Spatial Arrangement. *International Conference on Pattern Recognition*, (2000)
- [18] Egenhofer, M. and Franzosa, R.: Point-set topological spatial relations. *International Journal of Geographical Information Systems*. 5:(2) (1991) 161-174
- [19] Shariff, R., *Natural-Language Spatial Relations: Metric Refinements of Topological Properties*. 1996, University of Maine, Orono, ME.
- [20] Papadias, D., Kalnis, P., and Mamoulis, N.: Hierarchical Constraint Satisfaction in Spatial Databases. *Proceeding of the Annual Meeting of the AAAI*, Orlando, FI (1999)
- [21] Michalewicz, Z.: *Gentic Algorithms + Data Structures = Evolution Programs*. 2nd ed. Springer Verlag, New York (1994)
- [22] Craenen, B., Eiben, A., and Marchiori, E.: How to handle Constraints with Evolutionary Algorithms. in Chambers, L. (ed.): *The Practical Handbook of Genetic Algorithms, Applications*, Chapman&Hall/CRC, Boca Raton London New York Washington, DC., (2001) 341-362
- [23] Eiben, A., Raué, P.-E., and Ruttkay, Z.: Solving Constraint Satisfaction Problems Using Genetic Algorithm. *1st IEEE Conference on Evolutionary Computation*: IEEE Computer Society (1994)
- [24] Eiben, A., Raué, P.-E., and Ruttkay, Z., *Heuristic Genetic Algorithms for Constrained Problems, Part I: Principles*. 1993, Vrije Universiteit: Amsterdam
- [25] Goldberg, D.: *Genetic Algorithm in Searching, Optimization, and Machine Learning*. Addison Wesley, (1989)
- [26] Guttman, A.: R-Trees: A Dynamic Index Structure for Spatial Searching. *ACM SIGMOD Int. Conf. on Management of Data*, (1984)
- [27] Blaser, A.: *Sketching Spatial Queries*. Ph.D. Thesis, Univedrsity of Maine, U.S.A, (2000)

Parallel Text Query Processing using Composite Inverted Lists *

Mauricio Marín

Computing Department, University of Magallanes, Chile
Center for Web Research, University of Chile (www.cwr.cl)
mmarin@ona.fi.umag.cl

Abstract. The inverted lists strategy is frequently used as an index data structure for very large textual databases. Its implementation and comparative performance has been studied in sequential and parallel applications. In the latter, with relatively few studies, there has been a sort of “which-is-better” discussion about two alternative parallel realizations of the basic data structure and algorithms. We suggest that a mix between the two is actually a better alternative. Depending on the workload generated by the users, the composite inverted lists algorithm we propose in this paper can operate either as a local or global inverted list, or both at the same time.

1 Introduction

In the last decade, the design of efficient data structures and algorithms for textual databases has received a great deal of attention due to the rapid growth of the Web [4]. Typical applications are those known as client-server in which users take advantage of specialized services available at dedicated sites by sending requests to one or more broker machines which in turn interact with the server [5]. In cases of very high traffic, efficient service time can only be achieved by distributing the user requests onto a set of computers or processors that work in parallel to serve the work-load.

This paper discusses the parallelization of inverted lists [4]; a popular data structure for supporting query processing in textual databases. We propose an efficient parallelization of inverted lists, which can be implemented using any communication library that supports distributed memory parallel processing. Its design combines the best of previous approaches: it mixes the so-called local and global index approaches into a single inverted list parallel algorithm which is more robust and efficient. This claim is supported by an analysis which is not tailored to particular implementations, benchmarks, or architecture of the parallel machine.

Algorithmic design and comparative evaluation is effected on top of the bulk-synchronous parallel (BSP) model of computing [9, 12]. Previous approaches to the parallelization of inverted lists can be seen as pseudo-BSP algorithms. But unlike previous approaches (mostly based on the message passing model of computing) [3, 5, 8, 1, 11], we exploit the structured form of BSP computations to realize and cost opportunities for efficient performance.

For example, we have realized that the local and global approaches can be seen as just two particular instances of a general approach to the parallelization of inverted lists. The local inverted lists strategy does not scale well with the number of processors, whereas the global one is scalable but it can easily degenerate into a very inefficient strategy. Thus a strategy for really large scale parallel architectures should operate at some point between the two. We

*This work has been partially funded by research projects Millenium Nucleous Center for Web Research, Grant P01-029-F, Mideplan, Chile, and Fondecyt project 1020803.

are currently investigating the practical realization of this idea. Nevertheless, the proposed combination of global and local inverted lists can be considered as a simple and practical strategy which is suitable for medium size platforms such as clusters of PCs and distributed memory multi-processors systems operating under the “sharing nothing” model for main and secondary memory.

Simulation results show that the proposed composite inverted lists structure achieves better performance in all cases. Other researchers have shown that it is indeed feasible to achieve efficient performance with actual implementations of distributed inverted lists [3]. We thereby focus on evaluating comparative performance since components in charge of operations such as ranking of documents remain the same, namely they have identical running time costs across the different strategies. Indeed, none of the algorithms is radically different in terms of their computational running time costs. The difference comes from metrics like communication, synchronization and load balancing. Under the BSP cost model, these metrics are affected only by the algorithmic design and not by its particular implementation and execution environment. Thus our simulations are sufficient as they measure those metrics in accordance with algorithmic behaviour, and convenient as they allow us to explore cases beyond the available hardware and impose demanding workloads on real-life instances of large textual databases.

The remaining of this paper is organized as follows. Section 2 describes the BSP model of computing and section 3 presents the overall design of the parallel server and its interaction with one or more broker machines. Section 4 describes the inverted lists and presents an analysis of its performance. Section 5 describes the composite inverted lists and section 6 presents our conclusions.

2 The BSP model of computing

In the bulk-synchronous parallel (BSP) model of computing [12, 9], any parallel computer (e.g., cluster of PCs, shared or distributed memory multiprocessors) is seen as composed of a set of P processor-local-memory components which communicate with each other through messages. The computation is organised as a sequence of *supersteps*. During a superstep, the processors may perform sequential computations on local data and/or send messages to other processors. The messages are available for processing at their destinations by the next superstep, and each superstep is ended with the barrier synchronisation of the processors.

The total running time cost of a BSP program is the cumulative sum of the costs of its supersteps, and the cost of each superstep is the sum of three quantities: w , hG and L , where w is the maximum of the computations performed by each processor, h is the maximum of the messages sent and received by each processor with each word costing G units of running time, and L is the cost of barrier synchronising the processors. The effect of the computer architecture is included by the parameters G and L , which are increasing functions of P . These values along with the processors' speed s (e.g. mflops) can be empirically determined for each parallel computer by executing benchmark programs at installation time [9].

As an example of a basic BSP algorithm let us consider a broadcast operation which will be implicitly used in the algorithms described in the following sections. Suppose a processor “wants” to send a copy of P chapters of a book, each of size a , to all other P processors (itself included). A naive approach would be to send the P chapters to all processors in one superstep. That is, in superstep 1, the sending processor sends P chapters to P processors at a cost of $O(P^2(a + aG) + L)$ units of running time. Thus in superstep 2 all P processors have available into their respective incoming message buffers the P chapters of the book. An optimal algorithm for the same problem is as follows. In superstep 1, the sending processor sends just one *different* chapter to each processor at a cost of $O(P(a + aG) + L)$ units.

In superstep 2, each processor sends its arriving chapter to all others at a cost of $O(P(a + aG) + L)$ units. Thus at superstep 3, all processors have a copy of the whole book. That is, the broadcast of a large P -pieces a -sized message takes $O(P(a + aG) + L)$ cost.

The practical model of programming is SPMD, which is realized as P program copies running on the P processors, wherein communication and synchronization among copies is performed by ways of libraries such as BSPlib [13] or BSPub [14]. Note that BSP is actually a paradigm of parallel programming and not a particular communication library. In practice, it is certainly possible to implement BSP programs using the traditional PVM and MPI libraries. Optimizations such as message packing, relaxed and subset synchronization are possible [12, 14]. A number of empirical studies have shown that bulk-synchronous parallel algorithms lead to more efficient performance than their message-passing or shared-memory counterparts in many applications [9, 10].

3 Server-broker relationship

We assume a server operating upon a set of P identical machines, each containing its own main and secondary memory. We treat secondary memory like the communication network. That is, we include an additional parameter D to represent the average cost of accessing the secondary memory. Its value can be easily determined by benchmark programs available on Unix systems. The textual database is evenly distributed over the P machines. If the whole database index is expected to fit on the P sized main memory, we just assume $D = 1$.

Clients request service to one or more broker machines, which in turn distribute them evenly onto the P machines implementing the server. Requests are queries that are solved by using an index data structure distributed on the P processors. We assume that the index is implemented using an inverted list which, as described in the next section, is composed of a vocabulary (set of terms) and a set of identifiers representing all the documents that contain at least one of the words that are members of the vocabulary. The inverted list data structure enables the efficient retrieval of all identifiers for which a given term appears in the respective documents.

We assume that under a situation of heavy traffic the server is able to process batches of $Q = qP$ queries. Every query is composed of one or more vocabulary terms for which it is necessary to retrieve all document identifiers associated with them. Only the identifiers of the most relevant documents are presented to the user, namely those which more closely match the user information need represented by the query terms. For this, it is necessary to perform a ranking of documents. A widely used strategy for this task is the so-called vector model [4], which provides a measure of how close is a given document to a certain user query. We assume that the reader is familiar with this method and overall terminology [4].

Minimal broker In order to exploit the available parallelism we try to minimize the amount of sequential work performed by the broker machine. We restrict its functionality to receive user requests, distribute the queries onto the processors (uniformly at random), receive the best ranked documents (K in total) from the server, and pass them back to the user.

The two most basic operations related to providing answers to user queries are left to the parallel sever. That is, the retrieval of document identifiers and its respective ranking. Both operations are effected in parallel where the broker is responsible for scheduling those in a manner that keeps load balance of processors work as close to the optimal $1/P$ as possible. This is achieved by the combination of two strategies.

Firstly, the terms of the vocabulary are distributed uniformly at random onto the processors. This kind of strategy has proven to be a very effective tool for destroying correlation among the input data [12]; query terms in our case, with imbalance coming from the fact

that user preferences could cause that many terms be routed to the same processor. We use a hashing function on the term's characters for this purpose. This function is used to distribute the vocabulary's terms at index construction time and during the broker's term distribution process.

Secondly, for every query the broker chooses a server processor in which performing the ranking of documents. Later this processor sends the K best ranked ones back to the broker. As the terms of a given query (or set of queries), are likely to be located in different processors, the broker chooses such processor in a way that tends to maintain a good load balance with all other processors. That is, this processor is chosen in a way that attempts to evenly distribute all ranking tasks onto the P processors. This is effected by maintaining a counter for each processor. These counters keep the number of ranking tasks scheduled in every processor for queries which have not been completed yet. A ranking task for a given query is scheduled on one of the processors that are associated with their respective terms. From those, the processor with the smallest counter value is the selected one.

Thus the query terms are distributed uniformly at random by ways of the hashing function. The targets processors for these terms perform the work required to retrieve their associated lists of document identifiers. Then the lists associated with every query are routed to the processor selected for their final ranking. Lists associated with different queries are expected to be routed to different processors.

A pseudo-code for the steps executed by the broker is the following,

```
while(true)
{
    msg= rcvMessage(); // wait until it gets a new message.

    switch( msg.from() ) // where is the message from.
    {
        case USER: // new query arrival.

            // select a server processor for the final ranking of documents.
            proc= rankingProcessor(msg.Query()); // apply load balancing.

            foreach term in msg.QueryTerms()
            {
                term.ranker(proc); // let it know its ranker processor.
                serverProc= hashFunction(term.str()); // distribute evenly.
                sndMsg(serverProc,term); // send the term to its processor.
            }

            break;

        case SERVER: // server answer arrival.

            updateLoadBalancingCounters(msg.Query());

            // send back to the user the query's results.
            sndMessage(msg.user,msg.rankedDocumentsList);

            break;
    }
}
```

Bulk-synchronous parallel server The details of the operations executed by the parallel server are described in the next section as its design is conditioned by the inverted lists strategy being employed. In general, the server executes a never-ending sequence of supersteps

in which batches of q terms are processed in each processor along with the ranking of documents. The terms are routed to each processor by the broker as described above. In each superstep, the processing of a new batch is started. Depending on the kind of index strategy employed the processing of a given term plus the associated ranking of documents can take two or more supersteps to complete. Thus at any given superstep we can have several batches being processed, each at a different stage of execution. This is intended to achieve a good amortization of communication and synchronization costs.

4 Inverted Lists

For a collection of documents the inverted lists strategy can be seen as a vocabulary table in which each entry contains a term (relevant word) found in the collection and a pointer to a list of document's identifiers that contains such term. These lists are called *inverted-lists*. Thus, for example, a query composed of the logical AND of terms 1 and 2 can be solved by computing the intersection between the inverted-lists associated with the terms 1 and 2. The resulting list of documents can be then ranked so that the user is presented with the most relevant documents first (the technical literature on this kind of topics is large and diverse, e.g., see [4]). Parallelization of this strategy has been tackled using two approaches.

In the local index approach the documents are assumed to be uniformly distributed onto the processors. A local inverted-lists index is constructed in each processor by considering only the documents there stored respectively. We thus have P individual inverted-lists structures so that a query consisting of, say, one term must be solved by simultaneously computing the local sub-lists of document identifiers in each processor, and then producing the final global list from these P local sub-lists.

The BSP realization of the local index approach is as follows. Once the broker machine routes by ways of the hashing function a term w belonging to a query u to processor i , this processor broadcasts the term w to all other processors in the current superstep. Every processor does the same for each term they receive. In the following superstep, all processors scan their local inverted lists to obtain the sub-list of document identifiers for each term they received in the previous broadcast. These sub-lists are then sent to the processors acting as rankers. Thus if processor k happens to be the ranker for query u , the sub-list associated with term w in processor i along with the ones located in all other processors for term w , are routed to processor k . The same is effected for all other terms belonging to the query u so the processor k can perform the final ranking in the following superstep. The size of these sublists is reduced by performing a pre-ranking before sending them to their rankers. Also if two terms of query u happens to be in the same processor a pre-merging is performed (see [3] for this kind of optimizations). The whole process of a query u takes 3 supersteps to complete and send back the final list of document identifiers to the broker.

The second approach is the so-called global index. Here the whole collection of documents is used to produce a single inverted lists index which is identical to the sequential one. Then the T terms that form the global term table are uniformly distributed onto the P processors along with their respective lists of document identifiers. This is done by ways of the same hashing function employed by the broker. Thus, after the mapping, every processor contains about T/P terms per processor. In the local index case, each processor contains the same T terms but the length of document identifier lists are closely a fraction $1/P$ of the global index ones.

The BSP realization of the global index is as follows. Like the previous strategy, each term is routed to one server processor by the broker. For each term w belonging to a query u the inverted lists associated with terms of u are retrieved in their respective processors. Then these lists are sent to the ranker processor defined for the query u to then proceed in the next

superstep like the local inverted lists case. The whole process takes 2 supersteps to complete.

Average case analysis The BSP cost of the two strategies is the following. Suppose that the average length of the document identifier lists is W . For large textual databases we should expect $W \gg P$. We also should expect that the terms tables fit fully on their respective main memories. Document identifiers lists are supposed to be stored on secondary memory. We consider the cost from the broker machine to the instant at which this machine receives the results from the BSP server. We cost the communication between the two using the BSP cost model. Actually broker machines can well be replaced by processes located in the same BSP processors.

For the sake of simplicity, we assume that the execution of the vector model over a set of document identifiers demands a running time that is proportional to the size of the set. This is an under-estimation of this cost as it usually requires sorting and merging of large identifier lists. For $D = 1$, this under-estimation makes it harder to amortize the cost of communication and synchronization of processors. For $D > 1$ the dominant cost comes from secondary memory, which is proportional to the size of the identifier lists.

In the local index approach it is necessary to let each processor work on the same set of $Q = qP$ queries. It is thus necessary to perform a broadcast of Q queries so that all processors get a copy. Assuming just one broker machine and using the method described in section 2, this operation can be effected at cost $P(q + qG) + L$. However, in the proposed implementation of the BSP server what happens is that every processor retrieves q queries from its incoming queries queue, and broadcasts them to all other processors at $q + qPG + L$ cost. After this broadcast, the processors work on the determination of the document identifier lists for the $Q = qP$ queries, and send every list of average size W/P to their source processors so that they can compose the final list of average size W . The cost of this superstep is $Q(W/P)(D + G) + L = qWD + qWG + L$. Finally, in the last superstep, the final lists of size W are composed and sent to the, say, P broker machines or processes, which is effected at $Q(W/P)(1 + G) + L = qW + qWG + L$ cost. Thus the asymptotic cost of the local index strategy for P brokers is given by

$$qWD + q(P + W)G + L.$$

For the case of one broker machine, the above cost must be incremented in $P(qWG)$ units. Clearly, it is more efficient to have one broker (machine or process) per BSP processor, and let them route queries at random among the BSP processors. Randomness is convenient since it allows the achievement of good load balance in cases in which brokers are not expected to receive a similar number of queries from the Web.

The global index strategy is cost as follows. The distribution of qP queries from a single broker machine takes similar time to the broadcast of the previous strategy, namely $P(q + qG) + L$. In the proposed BSP server, once every processor gets its q queries, they build in parallel the identifiers lists for each query at a cost of qWD units, and then send the results at a cost of qWG units if P brokers are participating. Thus the total asymptotic cost is given by

$$qWD + qWG + L.$$

For real-life situations where $W \gg P$ (cases which justify the use of parallelism) the costs of the two approaches is essentially the same. Similar to the previous case, a term $P(qWG)$ must be included in the above expression when the results are sent to a unique broker machine. Also an additional superstep is needed in situations in which each term of a, say, two-terms query is processed in a different processor. In this case it is necessary to send one of the W -sized lists to the other processor at a cost of $WG + L$ units. All this makes the global index strategy more similar to the local index strategy.

An efficiency reduction problem in the global index strategy may arise when the most frequently visited terms tend to be located in the same processors. This produces load imbalance both in computation and communication. There exists some solutions to this problem. For example, in [5] a statistical analysis of terms co-occurrence in every document is effected at initialization time in order to determine which terms should be mapped on different processors. This is an example of static mapping. However, below we provide empirical evidence that when static mapping is done in a randomized manner by using a hash function this imbalance does not arise. On the other hand, a dynamic re-distribution of terms can be applied to move pairs (term, list) to other processors when load imbalance is detected.

The cost of the sequential inverted list strategy is simply $q P W D$ or it may well be $q P W D + q P (1 + W) G + L$ when either one or more broker machines are considered, and they are actually separated from the sequential server (we assume the former case for comparison purposes). Thus the comparative performance of parallel inverted lists can be severely limited by its communication costs when processing requests from just one broker machine, as its performance is bounded by $P (q W G)$.

Note that for systems with small W , we can expect a modest performance improvement of the global index over the local index approach. This is because the cost of broadcasts in the local index approach tends to be more significant for small W . However, real-life textual databases produce index structures with very differing lengths for the document identifier lists (e.g., try to evaluate the Zipf's formula described in [4]). Combining a term with a small-sized list with a term with a large-sized one into the same query can have a catastrophic effect in the global index approach. The local index approach does not have this problem but it is less efficient with small-sized lists because of the relative increase of the cost of broadcasts. This clearly suggests an approach which combines the two strategies. That is, terms with large identifier document lists are treated using the local index approach whereas terms with small lists are treated using the global one.

For the global index approach, we have realized that the initial mapping of the inverted lists structure onto the BSP processors can be made in a very simple and efficient manner. Once the whole sequential structure has been completed by using the usual procedure (the fully parallel construction of inverted lists has been investigated in [7]), we can pick up one by one the pairs (term, list), where "list" is the list of document identifiers where the "term" occurs in, and distribute them evenly onto the processors by selecting them uniformly at random. For this we can construct a hashing function on the term characters. The same function is used by broker machines during the routing of queries. A query with two or more terms is treated as a set of two or more one-term queries, and one of the BSP processors (uniformly at random as well) is given the task of composing the final document list for these many-terms queries.

Simulation study To motivate the introduction of the composite approach described in the next section, we present simulation results describing the performance of the two the local and global approaches to parallel inverted lists (document ranking is performed as in [3]).

Table 1 shows the performance of global and local inverted lists for 8 ... 64 processors. The results were obtained for 128 new query arrivals per superstep with simulation runs of 20000 completed queries. Queries were randomly generated by choosing uniformly from 1 to 4 terms, wherein terms for the first and second part of the table were also chosen uniformly at random from a total of 1300 and 6500 terms respectively. The sizes of the lists associated with these terms ranged from L_a (the maximum) to L_b (the minimum) in accordance with the Zipf's law normalized to the FR-TREC collection excluding stopwords as described in [2].

Efficiency columns E_e and E_m indicate the load balance (speed-up divided by the number of processors) for computation and communication respectively. The ratio m/e indicates the

P	L_a	L_b	glob. E_e	lists E_m	index m/e	local E_e	lists E_m	index m/e
8	116	76	0.88	0.88	0.68	0.92	0.93	1.38
16	116	76	0.82	0.79	0.73	0.89	0.89	1.52
32	116	76	0.75	0.70	0.73	0.82	0.85	1.69
64	116	76	0.62	0.56	0.75	0.76	0.79	1.92
8	104355	76	0.58	0.72	0.25	0.97	0.92	0.29
16	104355	76	0.47	0.52	0.35	0.91	0.81	0.45
32	104355	76	0.30	0.35	0.41	0.81	0.72	0.65
64	104355	76	0.19	0.18	0.52	0.58	0.50	0.83

Table 1: Global vs local inverted lists. 128 new queries per superstep

total amount of communication over the total amount of computation effected during the simulation.

The results of table 1 confirm the above claims, namely the global and local approaches can outperform each other under different situations. First and given that only 128 new queries arrive in each superstep, for 64 processors we have a modest amount of terms to be processed in each cycle (between 128 and 512, with average 320). Thus as the number of processors increases from 8 to 64 we see a clear reduction of the efficiencies in computation and communication. As expected, the local strategy has better efficiencies in all cases. Its efficiency is decreased only because of imbalance in the process of composing the final answers to queries. However, when the differences between inverted list sizes is small, the efficiencies of the global approach are competitive. In addition, for small sized inverted lists the efficiencies in both cases are fairly the same but the larger values of m/e for the local approach show that broadcasts start to have a significant effect in the communication cost. Note that queries using terms which are more specific are expected to work on small inverted lists. When we increase from 128 to 1024 the number of queries that arrive at every superstep, efficiencies increase to be near optimal. This for terms selected uniformly at random. However, if we chose terms with very large and very small inverted lists sizes to compose the queries, this in an alternate and random fashion, then the efficiencies decrease rather dramatically in the global lists approach. A situation like this can arise when users submit queries that contain very specific terms and less specific ones.

Note that in the proposed BSP server both the selection and ranking of documents is performed in the BSP processors. In particular, the ranking, which is running time demanding, is performed in parallel by attempting to choose a different processor to rank the documents associated with every query. For small rate of query arrivals per superstep, the way we select these processors can have a significant impact on load balance and thereby in computation and communication efficiencies. The load balancing method we described in section 3 is simple and allows the achievement of efficiencies much better than just selecting at random the ranker processor.

5 Composite Inverted Lists

It is straightforward to combine into a single BSP algorithm the two above described inverted lists strategies. Each processor maintains a hash table to keep information about which terms are kept as in the local index case and which as in the global one. The following pseudo-code shows the superstep executed by a BSP server using a composite inverted list index to solve user queries,

```

while(true) // Each BSP processor executes the same superstep.
{
    * Receive new messages and put them in a queue Q.

    * Foreach message msg in the queue Q do
    {
        switch( msg.type )
        {
            case BROKER: // new term received from the broker.
                if ( IsLocal(msg.term) == true )
                    Broadcast(msg.term);
                else
                { // retrieve and sub-rank document list.
                    List= getInvertedList(msg.term);
                    subList= preRanking(List);

                    // buffer message to be sent to the ranker processor.
                    bufferMsg(msg.ranker,RANKING,subList);
                }
                break;

            case BROADCAST: // arrival of a broadcast term.
                List= getInvertedList(msg.term);
                subList= preRanking(List);
                bufferMsg(msg.ranker,RANKING,subList);
                break;

            case RANKING:
                if ( queueSize(msg.queryId) == msg.numTermsQry )
                {
                    List= CalculateFinalRanking( dequeueAll(msg.queryId) );
                    bufferMsg( broker, SERVER, List);
                }
                else // queue up to wait for more terms
                    enqueue(msg.queryId,msg);
        }
    }

    * Send all buffered messages to their target processors,
    and synchronize processors.
}

```

A term is treated as global or local depending on the size of its associated inverted list. We set the maximum size of a list to be the one which produces the same ratio of computation to communication than the global inverted list approach. List sizes below this maximum are treated as in the global index case whereas sizes above the maximum are treated as in the local index one. Note that most queries containing relevant terms tends to have inverted lists of small sizes. Table 2 shows simulation results for the same conditions to the one above described. It can be seen that efficiencies are similar to those of the local approach whilst the ratio communication/computation (m/e) are similar to that of the global approach.

Bucket inverted lists A more general approach is to look at an intermedia situation between the local and global approaches. A realization of this case is to set buckets of a fixed size whose value is defined by the capacity of disk pages. Every inverted list is divided in a certain number of buckets which are distributed evenly onto the processor. The objective is to achieve a high degree of parallelism during the disk operations required to retrieve the

P	comp. E_e	lists E_m	index m/e	glob. E_e	lists E_m	index m/e	local E_e	lists E_m	index m/e
8	0.88	0.88	0.68	0.88	0.88	0.68	0.92	0.93	1.38
16	0.82	0.79	0.73	0.82	0.79	0.73	0.89	0.89	1.52
32	0.75	0.70	0.73	0.75	0.70	0.73	0.82	0.85	1.69
64	0.62	0.56	0.75	0.62	0.56	0.75	0.76	0.79	1.92
8	0.97	0.90	0.25	0.58	0.72	0.25	0.97	0.92	0.29
16	0.90	0.78	0.37	0.47	0.52	0.35	0.91	0.81	0.45
32	0.75	0.61	0.44	0.30	0.35	0.41	0.81	0.72	0.65
64	0.53	0.43	0.54	0.19	0.18	0.52	0.58	0.50	0.83

Table 2: Composite, global and local inverted lists. 128 queries per superstep. The first part of the table is for $L_a = 116$ and $L_b = 76$, and the second part is for $L_a = 104355$ and $L_b = 76$.

inverted lists associated to the terms of a given query. The distribution of buckets can be effected as follows. Suppose that the hash function used by the broker determines that a term w is to be routed to processor i and that the inverted list associated with w requires b buckets. Then the first block is stored in processor i and the following ones are stored in processors $(i + 1) \bmod P$, $(i + 2) \bmod P$, ..., $(i + b - 1) \bmod P$. Broadcasts are now processed considering the particular order of these consecutive buckets which allows one to improve its efficiency [9].

In addition, this strategy has the potential of reducing significantly the cost of broadcasts for queries with $b < P$ and thereby it scales up more efficiently than the local approach, whilst it has the same cost for $b = P$. The simulation results are similar to those of the local-global approach. The actual differences in performance become evident for very large number of processors. Note that we did not find that the computational overheads introduced by the implementation of this bucket inverted lists approach were not significant in terms of running time.

6 Conclusions

We have presented BSP algorithms for implementing global and local indexes devised to support efficient query processing in textual databases. Global refers to indexes built by considering the whole collection of documents, whereas local refers to indexes built by considering only a fraction ($1/P$) of the whole collection.

In the global one, the answer to a given query is constructed by only one processor, and in the local one the answer is constructed by all P processors in parallel. We considered cases that justifies the adoption of parallel computing, namely situations of very high traffic of queries continuously flowing into a server. In this context optimized the throughput of the overall system rather than particular queries. We propose a general BSP algorithm for implementing such a server.

For solving the problem of speeding up the solution to queries, we have studied the BSP realization of inverted lists. Their running time costs were expressed in terms of the BSP cost model in order to compare the different approaches to the parallelization of inverted lists upon the same framework. This provided us with running time expressions which are not tailored either to particular implementations of the algorithms or particular architectures of the parallel computer. This also allowed us to identify opportunities for significant optimizations of the total running time, mainly in the form of pipelining across supersteps. Also, the BSP cost model allowed us to perform simulations which focus on the same performance metrics, namely we focused on the most crucial aspects which affect the performance of actual implementations of the algorithms.

Overall, our main conclusions are that for inverted lists both the local and global index approach have essentially the same running time cost for inverted lists of fairly the same size. However, the performance of both approaches depends significantly on the behaviour of the work-load generated by the user queries. In particular, the queries mixing inverted lists of very differing sizes can have negative effects on the performance of the global inverted lists approach. On the other hand, queries producing small inverted lists can lead to a poor performance to the local inverted lists approach because of the relatively more significant cost of the periodical broadcast operations that it is necessary to perform. Clearly a combination of the two approaches is a solution which easily produces a practical and more robust strategy.

A generalization of the above strategies is bucket inverted lists strategy we described in this paper. Algorithms are a bit more involved and we believe this strategy is best suited for very large number of processors. We are currently studying the performance of this strategy along with its efficient implementation and load balancing method.

References

- [1] A. A. MacFarlane, J.A. McCann, and S.E. Robertson. Parallel search using partitioned inverted files. In *7th International Symposium on String Processing and Information Retrieval*, pages 209–220, 2000.
- [2] M.D. Araujo, G. Navarro, and N. Ziviani. Large text searching allowing errors. In *Workshop on String Processing (WSP'97)*, pages 2–20, 1997. Valparaíso, Chile (Carleton Univ. Press).
- [3] C. Santos Badue, R. Baeza-Yates, B. Ribeiro-Neto, and N. Ziviani. Concurrent query processing using distributed inverted files. In *8th International Symposium on String Processing and Information Retrieval*, pages 10–20, 2001.
- [4] R. Baeza and B. Ribeiro. *Modern Information Retrieval*. Addison-Wesley., 1999.
- [5] S.H. Chung, H.C. Kwon, K.R. Ryu, H.K. Jang, J.H. Kim, and C.A. Choi. Parallel information retrieval on a SCI-based PC-NOW. In *Workshop on Personal Computers based Networks of Workstations (PC-NOW 2000)*. (Springer-Verlag), May 2000.
- [6] W.F. McColl. General purpose parallel computing. In A.M. Gibbons and P. Spirakis, editors, *Lectures on Parallel Computation*, pages 337–391. Cambridge University Press, 1993.
- [7] B. Ribeiro, J. Kitajima, G. Navarro, C. Santana, and N. Ziviani. Parallel generation of inverted lists for distributed text collections. In *XVIII Conference of the Chilean Computer Science Society*, pages 149–157, 1998.
- [8] B.A. Ribeiro-Neto and R.A. Barbosa. Query performance for tightly coupled distributed digital libraries. In *Third ACM Conference on Digital Libraries*, pages 182–190, 1998.
- [9] D.B. Skillicorn, J.M.D. Hill, and W.F. McColl. Questions and answers about BSP. Technical Report PRG-TR-15-96, Computing Laboratory, Oxford University, 1996. Also in *Journal of Scientific Programming*, V.6 N.3, 1997.
- [10] D.B. Skillicorn and D. Talia. “Models and languages for parallel computation”. *ACM Computing Surveys*, 20(2):123–169, June 1998.
- [11] A. Tomasic and H. Garcia-Molina. Performance of inverted indices in shared-nothing distributed text document information retrieval systems. In *Second International Conference on Parallel and Distributed Information Systems*, pages 8–17, 1993.
- [12] L.G. Valiant. A bridging model for parallel computation. *Comm. ACM*, 33:103–111, Aug. 1990.
- [13] BSP Worldwide Standard, <http://www.bsp-worldwide.org/>.
- [14] BSP PUB Library at Paderborn University, <http://www.uni-paderborn.de/bsp>.

Section 11

Interactive Systems

This page intentionally left blank

Human Reasoning In Soft Computing

Vive S. Kumar

Simon Fraser University

2400 Central City, 10153 King George Highway, Surrey, BC, Canada

vive@sfu.ca, www.sfu.ca/~vivek

Abstract

Most contemporary computing systems situate humans only as the end users. Alternatively, human-in-the-loop computing systems employ humans to buttress the system when faced with hard tasks, which enables human reasoning to be used as part of the computational process. This paper discusses how human reasoning can be embedded as part of a Soft Computing environment and examines its implications. It exemplifies the arguments using a prototype help recommender system and highlights how pedagogical recommendations are made with the integration of human reasoning with system reasoning mechanisms. It also highlights a number of research fronts that result from embedding human reasoning to complement other reasoning mechanisms that are present in Soft Computing.

1. Introduction

Human intervention is pertinent when computational systems are faced with, among other issues, unmanageable processing complexity, inadequate failure handling, and deficient self-improvement. In such situations, the human-in-the-loop (HITL) [1,7,17] approach brings in humans to complement the system and attempts to buttress the system's computation leading to an optimal trade-off between human reasoning and system reasoning, where the trade-off is oriented towards satisfying the needs of the end user.

The success of the trade-off is dependent on the interface, communication protocols, and a shared understanding of the end user's goals between the human and the system. This paper introduces a soft computing technique that combines information obtained from three different AI reasoning mechanisms to provide the aforementioned optimal trade-off. The technique is implemented and tested in the domain of online help recommendation using a prototype system named Helper's Assistant (HA) [7].

The first section of the paper characterizes HITL and its relevance to Soft Computing. The second section describes the help recommender prototype, enumerates how HITL works in help systems, and explains how a Soft Computing (SC) technique attempts to resolve one of the key issues: pedagogy. The third section comments on the results and identifies key future directions.

2. Integrating human-in-the-loop in soft computing

HITL methodology employs humans as part of the computational process. There are two types of circumstances that merit HITL: first, the computational system not able to handle the task and hence require human support; second, the task is too difficult for the human to cope with and hence delegate a part of the task to semi-autonomous systems [4]. This paper focuses only on issues arising out of the first type of circumstances.

HITL establishes a bridge between systems attempting to meet the end user's goals and the human reasoning of the person involved in the computational process. It attempts to transfer human knowledge into computational models (mathematical and/or symbolic) and on the other hand attempts to convey computational reasoning to humans. Such transfers introduce hard system issues such as unmanageable processing complexity, inadequate

failure handling, and deficient self-improvement, which can be handled by SC techniques. The novel idea here is the introduction of human reasoning to complement other reasoning mechanisms using a HITL interface.

Unmanageable processing complexity of a task implies that the task is computationally complex, if not intractable, to be completed by the system to the expectations of the end user. Höök et al. [5] identify one such complex task in the domain of adaptive hypermedia. The task involves the representation of user characteristics and the customization of hypermedia interface in the dynamic and highly unstructured domain of information filtering. In light of the inadequacies of stereotype modelling and the difficulties in placing the right level of trust in autonomous systems, Höök et al. propose placing a human editor in-the-loop. The human editor is expected to combine his/her professional skills with machine intelligence in order to filter information and get feedback on user preferences.

In another scenario, the system can delegate the task to the human when the input received cannot be parsed. For example, the system could ask for the interpretation of the human if the end user's natural language input is not comprehensible to the system. Similarly, the system can delegate the task to the human when it fails to adequately handle underdeveloped solutions and/or errors that arise due to the fragility of the algorithm or when the system recognizes that the result is a 'local maxima'. An example of an underdeveloped solution is often observed in an online library system, where students fail to extract the necessary information from the system. The 'Ask Us Live' library services [13] of the Simon Fraser University supports students to find information from a human librarian using technology that combines Internet chat with co-browsing (the ability to share web sites, database searches, and other information between the librarian and the patron). HITL-oriented SC techniques could mediate students who fail to obtain the necessary library information from the online library resources to the 'Ask Us Live' services. In addition, these techniques could summarize, synthesize, and present the context of the end user (patron) interaction with the online library system to the librarian, ahead of time, to prepare the librarian for the forthcoming help session.

Yet another situation when the human can be brought into the processing loop is when the system recognizes its inability to sufficiently learn from its past experiences to cater towards its self improvement. Kecman [15] details how SC techniques are used to obtain analytical models using learning from experimental data (examples, measurements, patterns, or observations) and by means of embedding existing structured human knowledge into the models. Kecman proposes *regularization* as a means to find a solution to an *ill-posed* problem. In this scenario, humans can introduce generic constraints to restrict the solution space.

Likewise, there are many situations when the system becomes brittle and could use the services of the human who is in-the-loop. For example, HITL can associate the complexity of a task with the expected time frame of completion set by the end user. That is, the system can delegate the task and a summary of the ongoing computational process to the human who is in-the-loop if the given task is not completed by the system within the time frame set by the end user.

SC includes methods of reasoning in indeterminate situations, i.e. when both uncertainty and vagueness are present. It differs from conventional (hard) computing in that it is tolerant of imprecision, uncertainty and partial truth [12]. It utilizes a combination of reasoning mechanisms from probability reasoning, fuzzy reasoning, constraint reasoning, rule-based reasoning, neural network reasoning, evolutionary reasoning, and reasoning based on fractals and chaos theories, in a complementary fashion, to attempt solutions for complex non-linear tasks with imprecise, uncertain, and dynamically changing conditions

[16]. Bonissone [12] provides an excellent overview of how different reasoning mechanisms complement each other in a SC environment to achieve the end user's goals.

This paper contends that SC techniques could integrate human reasoning with system reasoning to provide generic solutions when the system becomes brittle. The next section exemplifies the arguments using Helper's Assistant (HA), a prototype help recommender system, and highlights how pedagogical recommendations are made with the integration of human reasoning with system reasoning mechanisms.

3. Integrating human and machine reasoning mechanisms in Helper's Assistant

Help recommender systems range from sophisticated graphical interfaces that guide users, to proactive help systems that can intervene in the user-software dialogue. They could be passive or active, provided with canned solutions or knowledge-based inferences, generic or task specific, collaborative or autonomous, centric or distributed.

Current help systems have a number of shortcomings: lack of context results in imprecise help solutions; lack of personalization results in unfocused help information; lack of a time-oriented response results in superfluous and insensitive help. These shortcomings motivate the integration of HITL-based human reasoning in SC in the domain of help systems with the goal of providing time-sensitive and personalized help recommendations [7].

HA is a peer-supported help recommender system. The current prototype works only in the domain of Java programming of the "string reversal" problem, where, as part of a programming assignment, first year students were asked to write a method to reverse an input string. Students could invoke HA when they had difficulties solving the "string reversal" problem. HA could identify a helper (the human who is in-the-loop) who was available online to discuss the difficulties with the student. Interestingly, HA presents a suite of relevant tools, task-specific suggestions, and pedagogical recommendations *to the helper* to supplement his/her interaction with the student. HA uses SC techniques to identify appropriate pedagogical recommendations to the helper.

3.1 Human-in-the-loop in Helper's Assistant

Human-in-the-loop in Helper's Assistant is made possible with the deployment of user models. A user model acquires, stores, and disseminates knowledge about users based on, among other information, the goals of the user, apriori knowledge, and inferred knowledge from stereotypes.

HA uses an extensive user model design that captures a variety of information referring to personal, relations, security, preference, performance, portfolio, collaboration, helping, diagnosis and other criteria related to the user. The captured information is represented in terms of many different data types including strings, numbers, objects, production-rules, constraints, hierarchical data, and graphical data. In addition, the model contains methods corresponding to each data value so that values can be accessed (get-) or subjected to a transformation (set-).

The first major task of the human-in-the-loop approach is the selection of a ready, able, and willing (RAW) human helper. A ready helper is one who is available online and logged on to HA. An able helper is one who is knowledgeable about the help request and related concepts. Ability of a helper can be decided based on the knowledge of the helpers as reflected in their user models. A willing helper is one who is open to receive help requests with specific task-specific constraints. The algorithm(s) to select RAW candidate helpers is dealt with elsewhere [3,10].

The second major task of the human-in-the-loop approach is extending support to the chosen helper in terms of a suite of relevant tools, task-specific suggestions, and pedagogical recommendations, which are accessible using an integrated interface. Help extended to the helper can be either non-intrusive (reactive) or intrusive (proactive). In a non-intrusive approach, the helper helps the student only when requested. In an intrusive approach, either the system and/or a set of helpers can monitor the progress being made by the student in completing a task, make helpful inferences, and intervene to offer help to the student.

The third major task of the human-in-the-loop approach is the ability to personalize help. Personalization focuses on the preferences of the helper. For example, helpers can specify their tool preferences in their user models. Help tools that are not preferred by the helper can be disregarded. Similarly, there are many places in HA where help resources are personalized to the requirements of the helper. See [7] for a complete description of the user model design and usage in HA.

The crux of the human-in-the-loop in help systems is the transfer of process control to a human helper when the system fails. On the other hand, SC enables the transfer of reasoning between the human and the system. Some of the key concerns relate to timing, protocol, communication, and outcome of such a transfer.

3.2 Pedagogical recommendations using soft computing techniques

HA employs a two-stage methodology to disseminate pedagogical knowledge to the helper. In the first stage, using a help context object [9] and pedagogical rules it selects a uninstantiated plan; in the second stage, it instantiates the plan using pedagogical constraints [8] that results in a suggested sequence of help steps/procedures.

In the first stage, HA uses a SC technique that combines fuzzy rule-based reasoning with probabilistic reasoning to select a uninstantiated plan. The pedagogical rules contain attributes that relate to the help context, the preferences of the helper (and the student), and the past interaction of the helper with HA. Presently there are twelve such attributes that are used in HA. The attributes address issues such as 'type of help-context', 'type of help response', 'student's competence', 'preferred help-principle', and 'time limits for resources'.

The pedagogical rulebase is based on Java Expert System Shell (JESS / FuzzyJESS) [2,11], where the pedagogical rules and the rule-based inference engine are embedded within the Java implementation of HA. An example pedagogical rule is given in Figure 1.

If	(student's expected help response = "debugging") and (student's competence in while loop = "low") and (student's preferred help principles include "maximum-to-minimum degree of help") and (helper's preferred help principle "unknown") and (student's previous use of "max-to-min degree of help" principle has been "successful") and (helper's earlier use of "maximum-to-minimum degree of help" principle has been "moderate")
Then	(use "maximum-to-minimum degree of help") and (update student's and helper's usage statistics)

Figure 1: Example pedagogical rule

The attributes can be represented in terms of fuzzy and non-fuzzy values. For example, variables like “helper's competence” can be added to the rule-antecedents with well-defined fuzzy values like “negative low”, “negative medium”, “negative high”, “nominal”, “positive low”, “positive medium”, and “positive high”. There are a number of predefined fuzzy sets in FuzzyJESS including *ZfuzzySet*, *TriangleFuzzySet*, and *SfuzzySet*, the use of which allows the system to accept feedback from the helpers in terms of fuzzy expressions.

The values for some of the attributes are obtained using a Bayesian reasoning system [18]. For example, a student's competence in a particular topic in Java can be estimated from the Bayesian belief network that represents the conceptual knowledge of the student in Java. The estimated probability value, say 0.3, can be mapped onto a fuzzy value, say ‘low’.

The rulebase is kept open for the inspection of the helper. That is, the helper can view the rulebase, modify the antecedents, or modify the values of the rule attributes, prior to running the engine. The rules are executed using JESS and the execution can be controlled with human reasoning. For example, limited by the time constraints, the helper can stop the execution of the rulebase in the middle and observe the intermediary values of the attributes. It is also possible for the helper to modify the intermediary values of the attributes and start/continue running the rulebase engine. While doing so, for instance, the helper can change the value of student's competence from ‘low’ to ‘moderate’ based on his/her observations and belief. Thus, HA embeds human reasoning within rule-based reasoning and provides an interface thus allowing the helper to control the degree of integration of the underlying reasoning mechanisms.

The result of the embedded rulebase-reasoning engine is handed over to the second stage, where a chosen uninstantiated help plan (also known as the help plan network) is traversed and instantiated using pedagogical constraints, resulting in a sequence of help steps/procedures.

Figure 2 presents a help plan network used in HA, where the nodes represent pedagogical actions (with associated resources) and the arcs represent the pedagogical constraints. The traversal is controlled by a constraint satisfaction reasoning mechanism, implemented using Java Constraint Library (JCL) [14]. There are different types of pedagogical constraints:

- constraints that establish prerequisite relations between two consecutive nodes,
- constraints that limit the number of execution of cyclic-nodes,
- constraints that ensure helper/student preferences are considered,
- constraints that guarantee the availability of help resources, and
- constraints that enforce time limits.

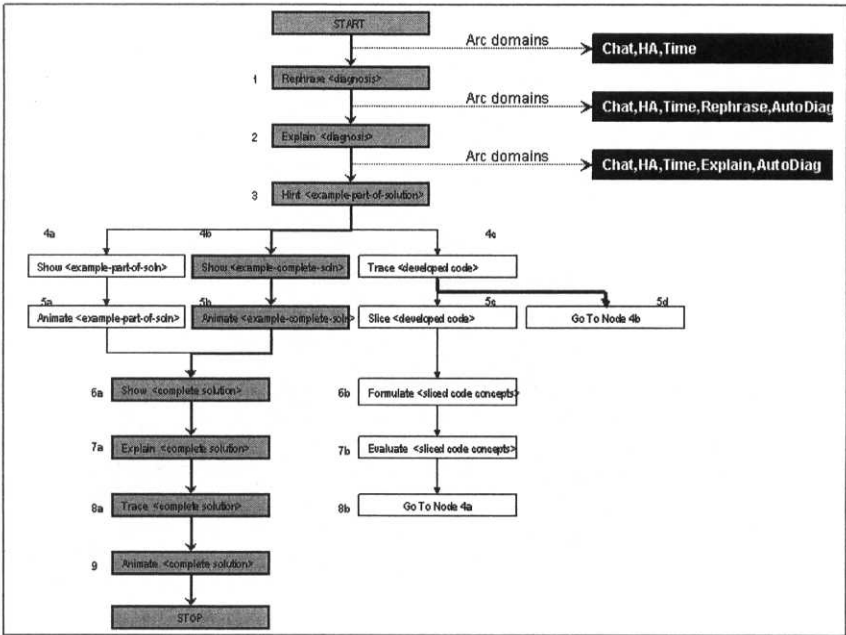


Figure 2: Help plan network with an instantiated path

Traversal of the uninstantiated help plan starts with the start node. It can continue through a path in the network by solving the pedagogical constraints attached to each arc. The execution of JCL can be halted temporarily after traversing each arc. At this time, the helper can halt or suspend the traversal process, or change the values of the constraints before allowing JCL to continue the reasoning process. In other words, HA enables human reasoning to be an integral part of constraint reasoning. However, this would work only in a network with limited number of nodes and when the number of values associated with the constraints is limited as well.

The current prototype of HA uses a Java interface to JCL. An interesting feature of the Java-JCL integration is the ability to instantiate the constraint domain values, variable values, and constraints at runtime.

In the current implementation, the variables and constraints are kept constant and only the domain values are instantiated for every invocation of the constraint solver. For example, the first arc is associated with domains 'Chat', 'HA', and 'Time'. For the 'Chat' domain, the prototype associates 'NetMeeting' and 'TalkTool' as the two domain values. The constraint resolution mechanism programmatically verifies the availability of these resources. For the 'HA' domain, 'HA server' and 'HA client' are the two domain values. For the 'Time' domain, the prototype associates a tuple, such as [1:30, 5:00], indicating a maximum wait time of 5 minutes and the remaining time (1:30) for which the end user is willing to wait. The maximum wait time is obtained from the student's user model and the remaining time is updated using the system clock.

The second arc contains two additional domains: 'Rephrase' and 'AutoDiag'. The domain values for 'Rephrase' are 'available' and 'not available', indicating the availability of a non-null database entry that explains the associated pedagogical action. 'AutoDiag' is yet another binary domain that indicates the availability of a database entry for the system diagnosis of the student's action. The current prototype dealt only with a well-defined Java

programming assignment and attempted to track the number of times each student compiled the code (within the HA interface) and also classified the syntax errors produced by the compiler for each compilation. The resultant values are presented to the helper as system diagnosis of student's conception of Java programming with respect to the programming assignment, the use of which allows the helper to determine the values of some of the domains.

The prototype implemented only two types of constraints: availability of resources and the availability of time. When resolved, these constraints may result in none, one, or more successor nodes. However, HA, prompted by the helper, selects only one of the successor nodes to be included in the traversal path. Likewise, the network is traversed until either the terminal node is reached or the constraints block the traversal to any other node or the helper halts the traversal. The path traced up to that point results in a candidate help-plan. Many candidate help-plans can be arrived at, in an automated or semi-automated fashion, which can be presented to the helper who can select an appropriate one for execution.

4. Conclusions and future directions

HA uses three complementary reasoning mechanisms: rule-based reasoning, Bayesian probabilistic reasoning and constraint reasoning. The rule-based reasoning mechanism is serially connected to the constraint reasoning mechanism, where the output of the former is used as the input for the latter. Both the reasoning mechanisms use Bayesian probabilistic reasoning, in parallel, to obtain values for some of the input. In addition, each reasoning mechanism embeds human reasoning that leads to the helper controlling the execution of the computational process as well as the reasoning process.

An informal usability study was conducted using an earlier version of the prototype to see how helpers responded with HA and without HA [7], while performing live help sessions with live students. The quality of help offered by helpers is compared between sessions with HA and sessions without HA, for the same set of help requests. Similarly, the quality of help offered by helpers is compared between expert helpers and novice peer helpers. The study resulted in many interesting observations based on the feedback from the pre-testers, the questionnaire filled out by the helpers and the students, recordings of audio and video of the helper-student interactions, recordings of the mouse clicks of the helper, and the notes of the experimenter. Some key observations are: helpers and students prefer short dialogues; novice helpers explored and looked for assistance within HA much more than experts did; novice helpers successfully answered questions more when HA was made available; and, both novice and expert helpers preferred integrated help environment. Based on these results, this paper claims that embedding human reasoning with system reasoning is a better strategy in online help systems when there are novice helpers.

Embedding human reasoning to complement other reasoning mechanisms that are present in a SC environment is a multifaceted challenge. First, the design of the interfaces for the reasoning mechanisms should be interoperable. For instance, JCL allows constraint reasoning to interface with Bayesian reasoning. In HA, the JCL constraints reasoning mechanism accepts the value for the subject knowledge of the helper from a Bayesian reasoning system. In other words, multiple reasoning mechanisms can co-exist in series or in parallel and human reasoning can be blended to various degrees in a hybrid Soft Computing system.

Second, reasoning mechanisms should be embeddable within each other. For example, fuzzy reasoning mechanism can seamlessly embed within rule-based inference mechanism as exemplified in Fuzzy JESS. However, there are mechanisms that are not naturally embeddable as can be seen between human reasoning and neural reasoning.

Third, semantics of the results of reasoning should be comparable. For example, as mentioned in Kecman and Pfeiffer [16], the underlying mathematical models of neural reasoning and fuzzy reasoning, respectively, are semantically similar. Such findings facilitate comparison of properties of the constituent reasoning mechanisms.

Fourth, the protocols of communication between reasoning mechanisms should be well defined. For instance, such protocols could specify when, how, and why a human intervene a reasoning process.

Fifth, task/domain/cognition – specific human knowledge representation and reasoning should be captured in mathematical or symbolic models, as exemplified in Sherlock [6], the use of which can help reduce the effects of ‘the curse of dimensionality’. However, the challenge is in modelling the compatibility and rigour (mathematical and/or symbolic) of human reasoning.

In addition, HITL introduces additional facets such as modelling the characteristics and preferences of the end user (including the end user’s expected time frame of completion of a task and the expected quality of the outcome), the estimation of the brittleness of the computational process, and the identification of the ready, able, and willing human (or humans) who can be brought into the loop to co-reason with the system.

SC is an emerging area in computational sciences that enables constituent reasoning techniques to co-exist and co-operate in a randomly changing domain with a goal to exploit tolerance for imprecision, uncertainty, and partial truth. SC is not just a resultant technology of combining reasoning mechanisms. The combination should be aimed and fine-tuned at addressing specific problems in the domain that are otherwise very hard to solve.

Kecman [15] states that human reasoning is not necessarily precise, quantitative, rigorous, and computable, but still it processes enormous amount of visual, acoustic, olfactory, tactile, and motor data, with the underlying ability to learn from experience, generalize from learned rules, recognize patterns, and make decisions. To do all these, humans must be employing an assortment of cross-functional and seamlessly interoperable reasoning mechanisms. The proposal to embed human reasoning as part of other reasoning mechanisms in SC is a step forward in the attempt to mimic natural intelligence using artificial, theory-oriented, and mathematically-rigorous models.

References

1. D.Tennenhouse. Proactive Computing. Available: <http://www.ttianguard.com/futuresystems/Proactive.pdf>.
2. Friedman-Hill E. (2000). JESS - The Java expert system shell. Available: <http://herzberg.ca.sandia.gov/jess/>.
3. Greer J.E., McCalla G.I., Kumar V.S., Collins J.A., & Meagher P. (1997). Facilitating collaborative learning in distributed organizations. Available: <http://www.oise.utoronto.ca/CSCL/papers/greer.pdf>.
4. I.Ivanisevic, & V.Iumelsky. (2000). Human Augmentation in Teleoperation of Arm Manipulators in an Environment with Obstacles. IEEE International conference on Robotics and Automation.
5. K.Höök, Å.Rudström, & A.Wærn. (1997). Edited Adaptive Hypermedia: Combining Human and Machine Intelligence to Achieve Filtered Information. Workshop in the Eighth ACM International Hypertext Conference. [Http://www.sics.se/~kia/papers/edinfo.html](http://www.sics.se/~kia/papers/edinfo.html).
6. Katz S., Lesgold A., Eggan G., & Gordin M. (1994). Modeling the student in Sherlock II. In Greer J.E. & McCalla G.I. (Eds.), *Student Modelling: The Key to Individualized Knowledge-Based Instruction* (pp. 99-125). Berlin, Germany: Springer-Verlag.
7. Kumar V.S. (2001). Helping the helper in peer help networks [PhD dissertation]. University of Saskatchewan.
8. Kumar V.S., Greer J.E., & McCalla G.I. (2000). Pedagogy in peer-supported helpdesks. International Conference on Knowledge-Based Computer Systems (KBCS 2000). Allied Publishers, New Delhi, India, 205-216.
9. Kumar V.S., McCalla G.I., & Greer J.E. (2001). Personalized contexts in help systems. 14th Canadian Conference on Artificial Intelligence (AI’01), 162-171.

10. McCalla G.I., Greer J.E., Kumar V.S., Meagher P., Collins J.A., Tkatch R., & Parkinson B. (1997). A peer help system for workplace training [In duBoulay B. and Mizoguchi R. (Editors), IOS Press: Amsterdam]. 8th World Conference on Artificial Intelligence in Education (AIED'97). Kobe, Japan, 183-190.
11. Orchard B. (2001). FuzzyJ ToolKit for the Java platform and Fuzzy JESS. Available: http://ai.iit.nrc.ca/IR_public/fuzzy/fuzzyJToolkit.html.
12. P.P.Bonissone. (1997). Soft computing: The convergence of emerging reasoning technologies. *Soft Computing*, 1(1), 6-18.
13. Simon Fraser University Library System. (2002). Available: <http://www.lib.sfu.ca/researchhelp/askus/index.htm>.
14. Torrens M. (2000). Java Constraints Library. Available: <http://liawww.epfl.ch/~torrens/JCL/>.
15. V.Kecman. (2001). *Learning and Soft Computing*. MIT Press.
16. V.Kecman, & B.-M.Pfeiffer. (1994). Exploiting the structural equivalence of learning fuzzy systems and radial basis function neural networks. Second European congress on Intelligent Techniques and Soft Computing (EUFIT-94), Aachen. Vol 1, 58-66.
17. Y.J.Tenney, & S.L.Spector. (2002). Comparisons of HBR Models with Human-in-the-loop Performance in a Simplified Air Traffic Control Simulation with and without HLA Protocols: Task Simulation, Human Data and Results. Available: <http://www.bbn.com/ambr/docs/tenney-spectorpaper.pdf>.
18. Zapata-Rivera J.D., & Greer J.E. (2000, June 19-23). Inspecting and visualising distributed Bayesian student models. The International Conference on Intelligent Tutoring Systems (ITS'00). Montreal. 544-553.

A Focus and Constraint-Based Genetic Algorithm for Interactive Directed Graph Drawing

Hugo A. D. do Nascimento^{*,†} and Peter Eades^{*}
 [hadrn,peter]@it.usyd.edu.au
 School of Information Technologies
 University of Sydney - Australia

Abstract. This paper presents a user-driven genetic algorithm for directed graph drawing. An interactive framework is considered where users can focus the algorithm on regions of the drawing that need major improvement, or include domain knowledge as layout constraints. The paper describes how focus and user constraints are managed by the genetic algorithm. The combination of user's skills with automatic tools allows a more flexible and efficient optimization method, when compared to traditional non-interactive genetic algorithms. Issues regarding memory usage, processing time, solution representation and convergence are discussed here.

1. Introduction

Drawings of directed graphs¹ appear in many different medias and applications, including science books, magazines, technical manuals, and software for designing, managing and exploring diagrams for simulation and tutoring purposes. When a graph contains only a few vertices and edges, it can be drawn easily by hand. However, as the number of vertices and edges increases, a manual drawing approach becomes very time consuming and difficult to manage. The solution to this problem is to make use of automatic techniques, which embed some user-desirable aesthetic criteria and apply a search method to find aesthetically pleasing drawings. A comprehensive study about aesthetic criteria and techniques for graph drawing is presented in [1].

For drawing of directed graphs, the most famous approach is the method of Sugiyama *et al.* [3], which consists of a four-step process involving several heuristics. There are also some approaches using meta-heuristics, in particular the genetic algorithms developed by [4] and [5]. In both cases, a directed graph is drawn as a hierarchy of horizontal lines called *layers*. The vertices of the graph are positioned in the layers so that a number of aesthetic criteria are satisfied. The criteria include: show a uniform orientation for the edges (e.g.: pointing downward), show few edge crossings, make the edges as straight as possible, and producing a drawing that is not too wide or tall.

^{*} Supported by the Australian Research Council.

[†] Lecturer of UFG-Brazil. PhD Scholarship from CAPES-Brazil.

¹ A graph $G=(V,E)$ describes a set V of entities, called vertices or nodes, and a set E of edges $e=(u,v)$ representing relationships between pairs of vertices of V . A graph is *directed* when its edges are oriented; (u,v) and (v,u) denote different oriented edges. We say that an edge $e=(u,v)$ is *connected* to u and v , and also that u and v are the *ending points* of e . For a complete terminology of Graph Theory, see [2].

Unfortunately, the satisfaction of these criteria involves several NP-hard problems, and only approximate solutions can be expected for many practical and interesting graphs. As an example, the heuristics developed by Sugiyama result in good drawings [1,3], but still with many edge crossings that could be easily removed. Genetic algorithm methods provide superior results, but their effectiveness decreases rapidly as the size of the graph grows [5]. For large graphs, genetic algorithms may also demand excessive processing time and, therefore, would not be suitable for interactive applications.

When automatic methods, even complex ones, fail in producing satisfactory drawings of graphs, no other alternative is left than to bring the human element back into consideration. In fact, many graph-drawing tasks are performed in two stages: an initial drawing is generated using some automatic tool, and then the user improves the drawing manually.

Here, we are interested in a more effective way of combining an automatic graph drawing method with human skills. Our aim is to allow a stronger interaction where the capabilities of both the human and the automatic method can be exploited. On one hand, automatic methods are usually fast; they can quickly explore several combinations of layouts and compute qualitative measures. On the other hand, users have subjective knowledge about aesthetics of diagrams, and seem to be good in visually identifying regions of a drawing with poor quality. In this research, we introduce a genetic algorithm for drawing directed graphs that can be driven by a user. The user and the algorithm are integrated in an interactive framework that allows focus on particular regions of the drawing for major improvement, inclusion of domain knowledge as layout constraints, and direct manipulation of the drawings. The main goal is to generate high quality drawings in less time.

This research is a natural continuation of our previous work [6], where we presented a version of the Sugiyama method modified for supporting focus and constraints. Experiments with the system showed that users could significantly improve automatically generated drawings of directed graphs, by reapplying the modified method.

The remainder of this paper is organized as follows: Section 2 describes our interactive framework for graph drawing and the system developed in our previous work. Section 3 presents the genetic algorithm with support for focus and layout constraints. Some issues regarding processing time, memory usage and convergence are discussed in Section 4. Section 5 draws our conclusions.

2. Interactive Framework

We draw a directed graph on an infinite rectangular grid of integer X - Y coordinates. The Y coordinates represent *layers*, while the X coordinates are called *columns*. Layers and columns are labeled with integer numbers. The labels start with any value, and increase continually from the bottom-most to the top-most layers, and from left-most to right-most column. A drawing of a directed graph G is created by assigning X - Y coordinates of the grid to all vertices of G . When an edge intersects one or more layers, it is said to be *long*. We insert a special vertex in each intersection point of an edge and a layer. This special vertex is called *dummy vertex* or *dummy node*. For differentiating the original vertices of the graph from dummy vertices, we called the former *real vertices* or *real nodes*. See Figure 1 for an example of a drawing of a graph on a grid. Real vertices are represented by rectangular boxes. Edges are shown as arrows connecting boxes. Dummy nodes are represented by small circles.

Formally, a drawing $D=(V,E,M,\delta)$ consists of a directed graph $G=(V,E)$, a list M of dummy vertices, and a function δ that assigns X - Y coordinates of the grid to every vertex v in $V \cup M$. Every dummy vertex is related to a particular edge e in E , and uniquely identifies

the intersection of e with a particular layer. We use the notations $v.l$ and $v.x$ to refer to the Y coordinate and the X coordinate, respectively, of a vertex v in $V \cup M$. If, for an edge $e=(u,v)$ in E , $|v.l-u.l|>1$, then e is *long* and has $|v.l-u.l|-1$ dummy vertices.

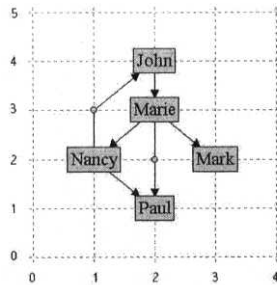


Figure 1: drawing of a directed graph on a grid.

The graph-drawing problem is to produce drawings that satisfy some aesthetic criteria such as those mentioned in the previous section. We consider an interactive framework where the user plays a very active role by helping an optimization method to improve an initial drawing of a directed graph. The interaction with the optimization method occurs by the user supplying *hints*, which are adjustments that may overcome a local minimum, reduce the space of solutions to be explored, or avoid ambiguity when there is more than one optimal solution. We consider three basic types of hints:

- **Focus.** The idea of focus is to identify regions of the drawing that need major improvement and concentrate the optimization method only on these. Since the complexity of managing a sub-problem is, in general, smaller than that of dealing with the entire problem, much processing effort may be saved. Focus is implemented by allowing the user to choose a group of vertices for redrawing. If a vertex is chosen, we call it *selected*; otherwise, it is called *fixed*. We focus the optimization method by redrawing the selected vertices, while preserving the X - Y coordinates of the fixed ones. The selection status of every vertex of the drawing D is kept in the drawing structure itself.
- **Constraints.** Constraints are useful for inserting domain knowledge into the problem after the optimization processing has been started. This is a common situation when the problem is dynamic and the user's preferences are subjective and hard to code. Constraints also allow solving ambiguity by restricting the structure of all feasible solutions. We use two types of constraints: Top-Down and Left-Right. They are modeled as special edges that define an ordering for a pair of real vertices of the drawing. A Top-Down constraint $c=(u,v)$ means that a vertex u has to appear above a vertex v in the drawing. Similarly, a Left-Right constraint $c=(u,v)$ means that u has to appear on the left-hand side of v . The user can add new constraints and remove existing ones.
- **Manual changes.** All optimization problems that are not solved by constraints and focus are managed in the framework by manual changes. In this case, the user directly changes the position of some vertices of the drawing. Such an operation is useful when the user visualizes changes that may lead to significant improvement of the drawing, but the optimization method does not implement it.

The interactive framework was implemented as a system called *GDHints*, where users can give hints in an intuitive way through a graphical interface. The system contains a visualization module that provides feedback to the users about the quality of the current

drawing being improved. The genetic algorithm presented in the next section is a graph-drawing optimization method that works in this environment. For more details about the *GDHints* system, see [6].

3. The Genetic Algorithm

The selected vertices of a drawing of a graph are the only elements that can be repositioned. The layout of the remaining part of the drawing must be preserved. At first glance, this may suggest isolating the selected vertices from the fixed nodes and redrawing only the selection. However, there is a strong dependency between these two groups of vertices, mainly due to edges connecting them. This situation is illustrated in Figure 2. Figure 2(a) is an initial drawing; vertices *d*, *e* and *f* are selected. Figure 2(b) is an improved drawing where the selected vertices were moved to a different position. The edges (*e*,*a*) and (*a*,*d*) forced these vertices to jump over the fixed vertex *c*, so that they could be placed on the left-hand side of the drawing, in a clearer configuration. Moreover, the fixed vertices limited the layout of the drawing since their positions could not be used by the selected vertices (assuming that vertex overlapping is not allowed). The total number of edge crossings was also taken into consideration when moving the selected vertices: it was given preference to a layout that minimizes edge crossings.

Due to the dependency between fixed and selected vertices, we cannot separate them completely. Therefore, another way of managing these elements is necessary.

Next, we outline how the selected and the fixed elements of a graph drawing are divided and represented in our genetic algorithm.

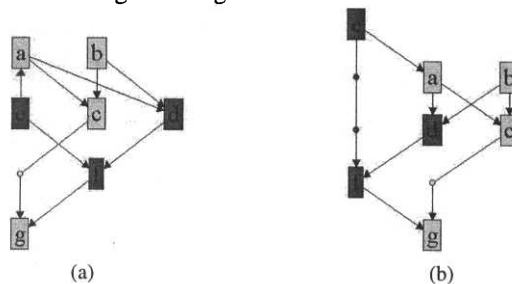


Figure 2: redrawn of selected vertices.

3.1 Individuals

We extend here the concept of selection in order to include edges: an edge of a drawing is *selected* if at least one of its ending points or dummy vertices (if they exist) is selected.

The genetic algorithm manipulates a drawing of the whole graph, a list of layout constraints and individuals. An *individual* contains information about only the selected vertices and the selected edges, and it can be used to reconstruct the selected area of the drawing. The genetic algorithm can have a *population* of individuals, each possibly describing a different layout for the selected area.

An individual consists of three main parts:

1. **REALV**: a fixed-length vector with the *X-Y* coordinates of all selected real vertices;
2. **EDGES**: a vector with one position for each selected edge of the drawing. Each position holds a list of all dummy vertices associated to its related edge. The list is sorted such that the dummy vertices are in the same order as they appear when

following the edge's orientation. Every dummy vertex in **EDGES** contains its X-Y coordinate position, whether this node is selected or not, and a reference to the edge to which it is related.

3. **DUMMYV**: a variable-length vector with references to the selected dummy vertices in **EDGES**. **DUMMYV** is used for quickly accessing the selected vertices only.

Note that an individual holds information about fixed dummy vertices as well as selected dummy vertices of a selected edge. Although fixed dummy vertices should not be redrawn, they have to be included in the individual since they can be deleted or introduced if their edges are lengthened.

An individual can be produced directly from a drawing D of the whole graph containing a selected set of vertices. We call this operation **Extraction**. As an example of **Extraction**, the drawing in Figure 2(a) would result in an individual with **REALV** containing coordinates for vertices d , e and f . The vector **EDGES** would have six positions – representing the selected edges (a,d) , (e,a) , (b,d) , (d,f) , (e,f) and (f,g) , each with empty lists, since there are no dummy vertices in these edges. The vector **DUMMYV** would be zero-length.

It is possible to combine a drawing of the whole graph with an individual; we name this operation **Merge**. The merge operation updates the sequence of dummy vertices and the coordinates of all selected vertices of a drawing according to the information in the individual.

3.2 Quality Evaluation

We define an evaluation function $Q(D,R)$ that measures aesthetic aspects of a drawing D , and the degree to which D satisfies a list R of constraints. This function results in a *cost vector*, whose parameters are as follows:

- q_1 : number of violated Top-Down and Left-Right constraints;
- q_2 : number of horizontal and upward edges;
- q_3 : number of edge crossings;
- q_4 : number of dummy vertices;
- q_5 : number of edge bends (a bend occurs when an edge turns to a different direction on a dummy vertex);
- q_6 : the area of the drawing (the number of layers multiplied by the number of columns used by the drawing); and
- q_7 : the sum of all edge lengths.

A priority order is defined such that q_i is more important than q_{i+1} for all $i=1,2,\dots,6$. This induces a lexicographic order on cost vectors. In general, low cost vectors result in aesthetically pleasing drawings and, therefore, in solutions of high quality.

Every individual produced by the system is assigned a cost vector. This is obtained by firstly merging the individual with an existing drawing of the whole graph, and then computing q_1, q_2, \dots, q_7 only for the selected vertices and edges. For instance, when computing q_3 we need to count the number of edge crossings between only selected edges and between a selected and a non-selected edge. The crossings between non-selected edges do not have to be considered for the cost vector, since they are an invariant in the problem.

3.3 Evolutionary Cycle

The genetic algorithm inputs an initial drawing D of a graph with some selected vertices, and a list R of Top-Down and Left-Right constraints. The algorithm outputs a

population of individuals representing different arrangements of the selected elements of the drawing.

The genetic algorithm executes the following steps:

1. An initial population P_0 is created by extracting an individual I from D , and applying the mutation operators to I several times in order to produce new individuals.
2. Repeat, for $iter=0$ until a stop condition is satisfied:
 - a. P_{iter} is evaluated by computing a cost vector for every individual in the population.
 - b. A sub-population S is selected from P_{iter} by tournament: the best individual in P_{iter} is inserted in S . The remaining individuals in P_{iter} take part in a number of pair-wise competitions: for any $I1$ and $I2$ chosen randomly from P_{iter} which have not yet participated in a competition, with cost vectors $Q1$ and $Q2$ respectively, if $Q1 < Q2$, then $I1$ is added to S ; otherwise, $I2$ is added to S .
 - c. A new population, P_{iter+1} , is created: the best individual in S is directly transferred to P_{iter+1} ; several mutated copies of this individual are also created and inserted into P_{iter+1} . Then two individuals $I1$ and $I2$ are randomly removed from S , modified by a crossover operator, and inserted into P_{iter+1} ; this process repeats until S is empty.

We use a stop condition based mainly on time.

The drawing D is necessary for computing cost vectors during the cycle, since it contains the dependency between the fixed and the selected vertices and edges. Moreover, the drawing provides the position of all fixed vertices. Such information is useful in order to avoid vertex overlaps when mutating or combining individuals.

3.4 Operators

This section describes the mutation and crossover operators implemented in our genetic algorithm. First, we introduce some basic routines.

3.4.1 Basic Routines

We developed five basic routines for helping to change the X-Y coordinates of selected vertices. The changes are applied to vectors REALV, EDGES and/or DUMMYV of an individual passed by parameter.

MOVEX (Individual I ; Vertex v ; Integer x). This routine changes the horizontal coordinate of a selected (real or dummy) vertex v in an individual I . First, it tries to move v to the position x in the same layer. If this position is already in use by another vertex, then the routine shifts some selected vertices or searches for an alternative horizontal position for the move. MOVEX needs to be carefully implemented to avoid overlap; see [7] for details.

MOVEY (Individual I ; Vertex v ; Integer y). MOVEY changes the vertical coordinate of a selected vertex v in an individual I to layer y , and calls MOVEX for solving overlaps.

UPCLOSURE (Individual I ; Vertex v). This routine returns a list of all selected vertices in layer $v.l$ or above, that can reach v , including v itself, following only edges with both ending points selected.

DOWNCLOSURE (Individual I ; Vertex v). This routine returns a list of all selected vertices in layer $v.l$ or below, that can be reached from v , including v itself, following only edges with both ending points selected.

MOVEYCLOSURE (Individual I ; Vertex v ; Integer y ; Boolean *upclosureflag*). This routine changes the vertical coordinate of a vertex v in I to layer y . Then it propagates the same amount of movement to all other vertices specified by *UPCLOSURE*(I, v) or *DOWNCLOSURE*(I, v), according to the parameter *upclosureflag*. *MOVEX* is called for solving overlaps for each vertex moved.

After executing *MOVEY* or *MOVEYCLOSURE*, all edges connected to the moved vertices are checked for invalid dummy nodes. If necessary, new dummy vertices are created and existing ones deleted.

3.4.2 Mutations

We implement three mutation operators:

RANDOMCHANGE (Individual I). This operator changes the coordinates of a selected vertex in an individual I . It randomly chooses a fixed or dummy selected vertex v , a direction to move (horizontal or vertical), and an offset $k \in \mathbb{Z}^*$. If the direction is horizontal or v is a dummy vertex, the operator calls *MOVEX*($I, v, v.x+k$); otherwise, it calls *MOVEY*($I, v, v.l+k$) or *MOVEYCLOSURE*($I, v, v.l+k, \text{False/True}$). *RANDOMCHANGE* is similar to the mutation *Shake* described in [5]. The main difference is that we set k according to a linear distribution of probabilities where small absolute values of the offset have a higher chance to be chosen. We also limit the offset, so that the area of the drawing is not increased much.

WALK_BC (Individual I). This operator follows the same idea of the mutation *Walk_BC* introduced by [5]; it implements a heuristic based on the famous barycenter method of Sugiyama *et al.* [3]. Our version of *Walk_BC* moves only selected vertices and considers both real and dummy nodes. It randomly chooses a selected vertex v and a vertical direction $d=1$ (for Up) or $d=-1$ (for Down). The X -coordinate of v in I is set to the arithmetic mean of all adjacent vertices of v on layer $v.l-d$. Then a vertex u adjacent to v in the direction d is chosen at random, and the barycenter process is repeated for u . The mutation continues walking until there is no more adjacent vertex to choose in the direction d . *WALK_BC* calls *MOVEX* in order to move selected vertices.

SOLVER (Individual I). This operator solves an unsatisfied constraint or changes the orientation of an edge that does not point downward. *SOLVER* randomly chooses a selected vertex u that is part of an unsatisfied constraint $c=(u, v)$. If c is a Top-Down constraint, then the operator calls *MOVEYCLOSURE*($I, u, v.l+1, \text{False}$); otherwise, if c is a Left-Right constraint, it calls *MOVEX*($I, u, v.x-1$). If there is no unsatisfied constraint, but the individual has an edge $e=(u, v)$ that is horizontal or upward, then the operator changes the Y -coordinate of u or v as if e were a Top-Down constraint. In the case that I does not violate any constraint and has no horizontal or upward edges, another mutation operator is called for changing I .

The mutation to be applied is chosen at random.

3.4.3 Crossovers

The genetic algorithm has one crossover operator:

COMBINE (Individual $I1, I2$). This randomly chooses a selected real vertex v that has different coordinates in $I1$ and $I2$, and swaps its positions in the individuals. *MOVEX* and *MOVEYCLOSURE* are used for swapping the coordinates of v .

3.5 Integration into the GDHints System

We have integrated the genetic algorithm into the *GDHints* system. The system provides an initial drawing of a graph using the Sugiyama method. The user then performs manual changes of the drawing, specifies layout-constraints, and/or selects some vertices for redrawing. The genetic algorithm was implemented as an independent processing thread that can be activated at any time. When activated, it creates a copy of the current drawing being visualized by the user and starts the evolutionary cycle. Every three seconds, the application locks the population of the genetic algorithm, recovers the best individual and merges it with the current (external) drawing. The user sees a progressive sequence of changes of the selected elements of the drawing when the best individual is continuously improved. Figures 3 and 4 show drawings produced by the *GDHints* system. The drawings on the left-hand side are initial layouts, generated by the Sugiyama method. The selected vertices are highlighted in a darker color. The drawings on the right hand-side are solutions produced by the genetic algorithm in 12 seconds (on a Pentium III 760Mhz running Windows Me).

The genetic algorithm works on practically any group of selected vertices. The user can focus on a single real vertex, a dummy node of an edge, several vertices, or all vertices of the graph. The selected vertices do not have to be adjacent in the graph or even in the drawing. Figure 4 shows improvements of two disjoint regions of a drawing.

A point to be noted is that, although the user can manually change the external drawing while the genetic algorithm is running, we do not propagate these changes to the algorithm. To do so, we would have to re-compute the cost vector of all individuals in the current population and possibly reconstruct the list of dummy vertices for some of the edges. Since this can be very time consuming, we prefer the user to stop the genetic algorithm in order to perform manual changes.

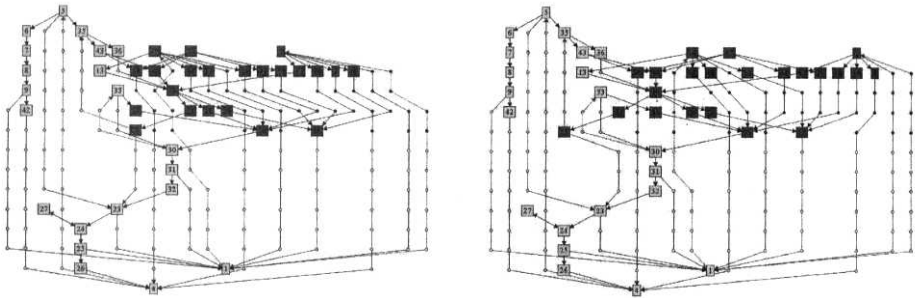


Figure 3: improvement of a selected graph drawing region.
(The Forrester's World Dynamics graph.)

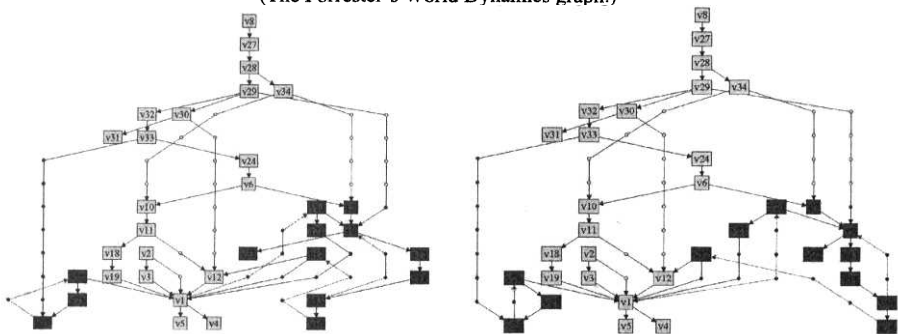


Figure 4: simultaneous improvement of two selected regions.
(C language syntax graph.)

4. Remarks

4.1 Time and Space

The structure of the individuals takes advantage of the focus mechanism in order to reduce memory usage and processing time. The individuals keep the minimal amount of information (the selected vertices and edges only) necessary to improve a region of a drawing. In addition, much processing time is saved by not evaluating the whole drawing when computing cost vectors.

The focus mechanism itself provides a reduction in processing time. The genetic algorithm is concentrated on an area of the drawing that needs major improvement, so that no time is spent in redrawing areas whose quality is already good.

Nevertheless, the merge process that integrates an individual into a drawing adds overhead to the genetic algorithm. If the whole drawing is selected, then our algorithm may take twice the time required by a non-focus genetic algorithm.

4.2 Solution Representation

The genetic algorithm in [5] draws only directed acyclic graphs. It implements a different solution representation, called *Edge Length Representation*, that stores relative coordinates for the vertices. This representation has the advantage of adjusting the position of several nodes automatically when moving a single vertex. In many cases, this effect helps the algorithm to escape from local minima.

Unfortunately, we cannot use the *Edge Length Representation* in our approach, since we also handle graphs with cycles. As an alternative, we use absolute *X-Y* coordinates in the individuals, and compensate the adjustment effect of the *Edge Length Representation* by having vertex-overlap resolution and closure-based movements implemented in the basic routines.

4.3 Solution Quality

We experimented with different setups for the genetic algorithm, and found that some options result in convergence to better solutions than others. For instance, the mutation *RANDOMCHANGE* without the linear distribution of probabilities (all offsets d have the same probability of being chosen) produces individuals with much lower quality. These individuals are difficult to improve and rarely propagate their characteristics to new generations.

The mutation *SOLVER* also gives better solutions. It provides a fast way of solving constraints and minimizing the number of upward and horizontal edges, which are the aspects with highest priorities in the quality measure of an individual.

4.4 Other Improvements

Optimization of the data structure becomes an essential issue if the genetic algorithm is going to be used in an interactive environment, where the response time has to be short. We have implemented some simple optimizations:

- 1) We keep track of which vertices and edges of the individuals are affected by a mutation or crossover operator. When computing the quality of an individual, we recalculate only the contribution of the affected elements to the cost vector.
- 2) A bucket structure is created for the drawing and for every new individual. The structure classifies all vertices by their layers, so that a list of the nodes in a particular layer can be efficiently recovered.

5. Related Work

Other researchers have explored different ideas for user interaction. Mendonça [8] presented an approach for tuning a simulated annealing for graph drawing: a machine learning system analyzes the quality of user-generated drawings and adjusts a set of aesthetic criteria and annealing parameters. Rosete-Suarez *et al* [9], Jacobsen [10], and Barbosa and Barreto [11] investigated a similar approach where the learning process involves user-ranking of drawings generated by a genetic algorithm. These works follow an orientation different from ours, but they are good examples of how user interaction can help meta-heuristics for graph drawing problems.

6. Conclusion

We introduced a new genetic algorithm for drawing directed graphs. The main differences to previous genetic algorithms are the support for focus and layout constraints, and the optimizations for working in an interactive environment. Users can focus the genetic algorithm in order to help convergence.

We also showed that the selected and fixed regions of a drawing are strongly linked, and presented a way of managing these two regions in the algorithm.

We are now exploring new configurations of the genetic algorithm and designing experiments for testing focus and layout constraints interactively.

Acknowledgements

We would like to thanks Jürgen Branke, from the University of Karlsruhe, Germany, for the valuable discussions about genetic algorithms and human-computer interaction.

References

- [1] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph drawing: algorithms for the visualization of graphs*. New Jersey: Prentice-Hall, 1999.
- [2] J. A. Bondy and U. S. R. Murty. *Graph Theory with applications*. Macmillan: London, 1976.
- [3] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical systems. *IEEE Trans. Syst. Man Cybern.*, SMC-11(2):109-125, 1981.
- [4] L. J. Groves, Z. Michalewicz, P. V. Elia, and C. Z. Janikow. Genetic algorithms for drawing directed graphs. In *Proceedings of the Fifth Int. Symposium on Methodologies for Intelligent Systems*, pages 268-276. Elsevier North-Holland, 1990.
- [5] J. Utech, J. Branke, H. Schmeck, and P. Eades. An evolutionary algorithm for drawing directed graphs. In *Proceedings of the 1998 International Conference on Imaging Science, Systems, and Technology (CISST'98)*, pages 154-160, 1998.
- [6] H. A. D. do Nascimento, and P. Eades. User hints for directed graph drawing. In *Proceedings of the 9th Graph Drawing Conference*, Lectures Notes on Computer Science, vol. 2265, pages 205-219, 2001.
- [7] H. A. D. do Nascimento, and P. Eades. A Focus and Constraint-Based Genetic Algorithm for Interactive Directed Graph Drawing. Technical Report TR533, ISBN 1-86487-512-7, School of Information Technologies, University of Sydney, Set 2002.
- [8] C. F. X. de Mendonça N. A layout system for information system diagrams. PhD Thesis, Department of Computer Science, University of Queensland, 1994.
- [9] A. Rosete-Suarez, M. Sebag, and A. Ochoa-Rodriguez. A study of evolutionary graph drawing. Technical Report, Sep 1999. URL: citeseer.nj.nec.com/411217.html.
- [10] A. E. Jacobsen. Interaktion und Lernverfahren beim Zeichnen von Graphen mit Hilfe evolutionärer Algorithmen ("Interaction and learning methods for graph layouts with the help of evolutionary algorithms"). Master Thesis, Institute AIFB, University of Karlsruhe, Germany, 2001.
- [11] H. J. C. Barbosa and A. M. S. Barreto. An interactive genetic algorithm with co-evolution of weights for multiobjective problems. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 203-210, Morgan Kaufmann Publishers, San Francisco, California, July 2001.

Dialogue Act Connectionist Detection in a Spoken Dialogue System*

Emilio Sanchis María José Castro

Departament de Sistemes Informàtics i Computació

Universitat Politècnica de València

Camí de Vera s/n, 46022 València, Spain

{esanchis,mcastro}@dsic.upv.es

Abstract. We present an approach to dialogue act detection within the framework of a domain-specific dialogue system. The task consists of answering telephone queries about train timetables, prices and services for long distance trains in Spanish. In this system, the representation of the meaning of the user utterances is made by means of *dialogue acts*, which determine the type of communication of the user turn, and by their associated *case-frames*, which supply the data of the utterance.

We focus on the classification of a user turn given the utterance in a specific class of dialogue act by using multilayer perceptrons. This classification can help in the posterior processes of understanding and dialogue management. Results of experiments with the correct transcriptions of the user utterances (text data) and with the sequences of words obtained from the recognition process (speech data) are presented.

1 Introduction

Spoken dialogue systems are a natural interface between users and machines in the framework of information systems. Due to the difficulty of the development of systems of this kind, different modeling and decoding processes must be explored. In particular, the contribution of artificial neural networks can be important in some parts of a dialogue system, such as in acoustic phonetic modeling [1, 2], language modeling [3, 4], or language understanding [5].

One of the most important problems in dialogue systems is the correct understanding of the user turns. Differently from speech recognition, whose objective is to find the correct sequence of words given an utterance, in the case of a dialogue system, the goal of the understanding module is to find the information supplied in the utterance and the purpose of the user turn. This allows us to ignore some useless words, but it is necessary to extract the relevant information to find the meaning. The work that we present in this paper is an approach to speech understanding by means of multilayer perceptrons. Following the natural use of neural networks, the problem is posed as a classification problem of the user turn in terms of some classes which were previously defined. To do that, the turns of a dialogue have a set of elemental units called *dialogue acts* [6, 7] associated to them. A dialogue act represents the intention of the turn or a part of the turn (that is, a turn can have more than one dialogue act associated to it). This kind of representation is specially interesting for the development of

*This work has been partially supported by the Spanish CICYT under contract TIC2002-04103-C03-03.

stochastic models of the dialogue structure [8]. On the other hand, a successful representation of the meaning of the sentences is to use *case-frames*. A case-frame represents not only the purpose of the sentence, but also the attributes (or values) supplied in it.

As mixed-initiative spoken dialogue systems can have many sources of errors (recognition errors, unexpected answers, etc.), it can be useful to have a way to reliably detect what kind of utterance has been pronounced. Once the kind of utterance (dialogue act) is determined, the attributes and values (case-frames) can be extracted in a posterior process with the advantage that the scope of the analysis is strongly restricted. We have focused our work in the definition and detection of the dialogue acts by using artificial neural networks. The dialogue task consists of an information retrieval system by telephone about train timetables, prices and services. Results of experiments with the correct transcriptions of the user utterances (text data) and with the sequences of words obtained from the recognition process (speech data) are presented.

2 The Dialogue Structure

One of the most common ways to represent the structure of a dialogue is by means of the so-called dialogue acts. The definition and the correct identification of dialogue acts is specially useful for obtaining a dialogue model and helping in the recognition and understanding of the utterances of the user. The task consists of answering telephone queries about train timetables, prices and services for long distance trains in Spanish. A set of labels was defined after the observation of 215 dialogues obtained by using the Wizard of Oz technique. The dialogues correspond to four different scenarios, which are queries about single and round trips, prices and one free scenario. In order to be able to work with labels that represent the general purpose of the turn, and with more detailed labels that represents more specific semantic information, a set of three-level labels was defined:

1. The **first level** is general for any task. It comprises the following labels:

<i>Opening</i>	<i>Closing</i>	<i>Undefined</i>	<i>Not_Understood</i>
<i>Waiting</i>	<i>Consult</i>	<i>Acceptance</i>	<i>Rejection</i>
<i>Question</i>	<i>Confirmation</i>	<i>Answer</i>	

2. The **second level** is specific to the task and represents the semantics of the utterance.

<i>Departure_time</i>	<i>Return_departure_time</i>	<i>Arrival_time</i>	<i>Return_arrival_time</i>
<i>Price</i>	<i>Return_price</i>	<i>Length_of_trip</i>	<i>Train_type</i>
<i>Return_train_type</i>	<i>Services</i>	<i>Confirmation</i>	<i>Not_Understood</i>
<i>Affirmation</i>	<i>Rejection</i>	<i>Closing</i>	<i>New_data</i>

3. The **third level** takes into account the values supplied in the utterances, such as the names of the cities, the dates, etc. . .

The corpus was labeled by means of a semiautomatic procedure. That is, some dialogues were manually labelled and the rest of the corpus was automatically labelled using the preliminary models obtained with these samples. Then a manual correction of the labels was done.

<i>Original sentence:</i>	Quería saber los horarios del Euromed Barcelona-Valencia. [I would like to know the timetables of the Euromed train from Barcelona to Valencia.]
<i>1st level:</i>	<i>Question</i>
<i>2nd level:</i>	<i>Departure_time</i>
<i>3rd level:</i>	<i>Departure_time</i> (<i>Origen:</i> barcelona, <i>Destination:</i> valencia, <i>Train_type:</i> euromed)

<i>Original sentence:</i>	Hola, buenos días. Me gustaría saber el precio y los horarios que hay para un billete de tren de Barcelona a La Coruña el 22 de diciembre, por favor. [Hello, good morning. I would like to know the price and timetables of a train from Barcelona to La Coruña for the 22nd of December, please.]
<i>1st level:</i>	<i>Question</i>
<i>2nd level:</i>	<i>Price, Departure_time</i>
<i>3rd level:</i>	<i>Price</i> (<i>Origen:</i> barcelona, <i>Destination:</i> la_coruña, <i>Departure_time:</i> 12-22-2002) <i>Departure_time</i> (<i>Origen:</i> barcelona, <i>Destination:</i> la_coruña, <i>Departure_time:</i> 12-22-2002)

Figure 1: Example of the three-level labeling for two user turns. The Spanish original sentence and its English translation are given.

One of the good characteristics of our approach is that a dialogue turn can be labelled by more than one label. This has the advantage that the labels represent more specific meanings of the sequences of words, but it contrast, finding the correct segmentation of the sentence is more difficult.

An example of the three-level labelling for some user turns is given in Figure 1. We will focus our attention on the second level of the labels (from now on, we will call them *frames*), which are used to guide the understanding process. Note that each user turn can be labelled with more than one label (see the second example in Figure 1).

3 Artificial Neural Networks for Language Understanding

Language understanding tasks have usually been based on symbolic architectures, which use explicit rules that operate on symbols [9]. In contrast, machine learning techniques for inferring structural models have also been applied to this field. Specifically, hidden Markov models and stochastic regular grammars have been successfully used in the understanding module of dialogue systems [10, 11].

Recently, artificial neural networks have been used in language understanding, but most of the connectionist language models implemented until now have had severe limitations [3, 4]. Understanding models have been limited to simple sentences with small lexica (see [5] for a revision of understanding and production neural network models).

In our approach, we used multilayer perceptrons (MLPs) for detecting the dialogue acts, which are strongly related to the semantics of the sentence (the frames). The number of input units was set by the size of the lexicon of the restricted-semantic task. There was one output

unit corresponding to each class to classify the sentences by their meaning.

3.1 Input to the MLP: Lexicon and codification of the sentences

A characteristic of some understanding problems is that, in many cases, it is not important to consider the differences among different morphologic variability of words (gender, number, verb tenses, articles, ...). This allows us to define categories and lemmas and thereby reducing the size of the vocabulary. In our task, we have the following categories:

1. *General categories*: city names, cardinal and ordinal numbers, days of the week, months.
2. *Task-specific categories*: departure and arrival city names, train types.
3. *Lemmas*: verbs in infinitive, nouns in singular and without articles, adjectives in singular and without gender.

This way we reduced the size of the lexicon from 637 to 311 words. Finally, we discarded the words that had a frequency lower than five, thus obtaining a lexicon of 138 words. Note that sentences which contained these words are not eliminated from the corpus, only the words were deleted. An example of the preprocessing of the original sentences is illustrated in Figure 2 (see *Original sentence* and *Preprocessed sentence*).

We do not consider that the sequential structure of the sentence is fundamental for detecting the type of dialogue act for this task.¹ For that reason, the words of a sentence were all encoded with a local coding: the input of the MLP was formed by 138 units, one for each word of the lexicon. When the word appeared in the sentence, its corresponding unit was set to 1; otherwise, its unit was set to 0. An example is given in Figure 2 (see from *Original sentence* to *Input local codification*).

3.2 Output of the MLP: Multiple labels

A total of 16 different dialogue act labels were defined for the task (see second level labels of Section 2). Each user turn could be labeled with more than one label (as in the second example in Figures 1 and 2). Thus, the desired outputs for each training sample were set to 1 for those (one or more) classes that were correct and 0 for the remainder.² We discarded the turns that were labeled with a class that had a frequency lower than five (a total of 1,338 user turns were selected), which comprised only 11 original labels:

<i>Departure_time</i>	<i>Return_departure_time</i>	<i>Arrival_time</i>	<i>Price</i>
<i>Train_type</i>	<i>Confirmation</i>	<i>Not_understood</i>	<i>Affirmation</i>
<i>Rejection</i>	<i>Closing</i>	<i>New_data</i>	

An example of codification of the frames is illustrated in Figure 2.

¹Nevertheless, the sequential structure of the sentence is essential in order to *segment* the sentence into slots to have a real understanding of the sentence.

²In [12], we performed two types of classification experiments (with text data): a maximum a posteriori approach (with extended frames) and a multiple a posteriori approach (with multiple frames). Better performance was obtained using this latter approach.

4 The Classification Problem

In this work, we focus on the classification of a user turn given in natural language (using categories and lemmas) in a specific class of frame (or classes of frames). Multilayer perceptrons are the most common artificial neural networks used for classification. For this purpose, the number of output units is defined as the number of classes, C , and the input layer must hold the input patterns. Each unit in the (first) hidden layer forms a hyperplane in the pattern space; boundaries between classes can be approximated by hyperplanes. If a sigmoid activation function is used, MLPs can form smooth decision boundaries which are suitable for performing classification tasks [13]. The activation level of an output unit can be interpreted as an approximation of the a posteriori probability that the input pattern belongs to the corresponding class.

As we desire to test multiple outputs (a user turn can have more than one dialogue act label associated to it), after training the MLP with multiple desired classes, an input pattern can be classified in the classes I^* with a posteriori probability above a threshold \mathcal{T} :

$$I^* = \{i \in C \mid \Pr(i|x) \geq \mathcal{T}\} \approx \{i \in C \mid g_i(x, \omega) \geq \mathcal{T}\},$$

where $g_i(x, \omega)$ is the i -th output of the MLP given the input pattern x and the set of parameters of the MLP ω . The set of classes C are the 11 labels defined in 3.2. The threshold \mathcal{T} is also learnt in the training process.

5 Experiments

We used multilayer perceptrons to detect the dialogue acts from the user turn (codified as explained in 3.1). The number of input units was set by the size of the lexicon of the task (138 words). There was one output unit corresponding to each class of dialogue act (the 11 classes defined in 3.2). The dataset (1,338 user turns) was randomly split into training set (80% of the user turns) and test set (20% of the user turns).

5.1 Training the MLPs

The training of the MLPs was carried out using the neural net software package “SNNS: Stuttgart Neural Network Simulator” [14]. In order to successfully use neural networks as classifiers, a number of considerations had to be taken into account, such as the network topology, the training algorithm, and the selection of the parameters of the algorithm [13–15]. Proofs were conducted using different network topologies of increasing number of weights: a hidden layer with 2 units, two hidden layers of 2 units each, two hidden layers of 4 and 2 units, a hidden layer with 4 units, etc. Several learning algorithms were also tested: the incremental version of the backpropagation algorithm (with and without momentum term) and the quickprop algorithm. The influence of their parameters such as learning rate or momentum term was also studied. Random presentation of the training samples was used in the training process. In every case, a validation criterion (20% of the training data was randomly selected for validation) was used to stop the learning process and to select the best configuration.

Table 1: MLP topologies and learning algorithms studied.

Topology:	One hidden layer: 2, 4, 8, 16, 32, 64 Two hidden layers: 2-2, 4-2, 4-4, 8-2, 8-4, . . . , 64-32, 64-64
Training algorithm:	Backpropagation (with and without momentum term), Quickprop
Learning rate:	0.05, 0.1, 0.2, 0.3, 0.4, 0.5
Momentum term:	0.1, 0.2, 0.3, 0.4, 0.5
Quick rate:	1.75, 2

5.2 *Selecting the best configuration of MLP*

During this training process, we first tested the influence of the topology of the MLP. We trained different MLPs of increasing number of weights using the standard backpropagation algorithm (with a sigmoid activation function and a learning rate equal to 0.2), selecting the best topology according to the mean square error (MSE) of the validation data. The minimum MSE of the validation data was achieved using an MLP of one hidden layer of 32 units.

We followed our experimentation with MLPs of this topology, training MLPs with several algorithms: the incremental version of the backpropagation algorithm (with and without momentum term) and the quickprop algorithm. Different combinations of learning rate and momentum term as well as different values of the maximum growth parameter for the quickprop algorithm were proved (see Table 1). The best result on the validation data was obtained using the MLP trained with the standard backpropagation algorithm and a value of LR equal to 0.3.

6 **Final experiment: Testing the best MLP with speech data**

Once we had selected the best combination of topology, learning algorithm and parameters for the MLP, according to the classification rate of the validation data, we proved the trained MLP with the test (*written*) data, obtaining a percentage of classification equal to 92.54%.

At this point, we wanted to prove the robustness of our system with speech data. To do that, we recognized the test utterances with a recognition system based on hidden Markov models [16]. The word accuracy for these utterances was 81.09%.

Then, the recognized sentences (with errors from the speech recognition module) were classified using the trained MLP, obtaining a percentage of correct classification equal to 72.39%. That is, when speech data is used, we get a 20% (approximately) of worsening: from 92.54% of classification with text data to 72.39% of classification with speech data. This worsening is related to the recognition errors, which also are in a range of 20% word error rate.

7 **Conclusions**

This study shows that, even with the automatically recognized sentences from the utterances, using a connectionist approach is very effective in the detection of dialogue acts.

This automatic process is helpful to the understanding module of the dialogue system: firstly, the speech recognition module supplies the recognized sentence from the user utterance; secondly, the MLP detects the dialogue act associated to the sentence; thirdly, a specific

understanding model for each dialogue act is used to segment and fill the case-frames. If the utterance is recognized and the dialogue act is detected with a high level of confidence, we could use a specific understanding model for the frame class. If it is not, a global understanding model could be used. Moreover, a combination of both types of models could be used to improve the understanding process in the spoken dialogue system.

References

- [1] H. Bourlard and N. Morgan. *Connectionist speech recognition—A hybrid approach*, volume 247 of *Series in engineering and computer science*. Kluwer Academic, 1994.
- [2] M. J. Castro and F. Casacuberta. Hybrid connectionist-structural acoustical modeling in the ATROS system. In *Proceedings of the 6th European Conference on Speech Communications and Technology (Eurospeech'99)*, volume 3, pages 1299–1302, Budapest (Hungary), September 1999.
- [3] Wei Xu and Alex Rudnicky. Can Artificial Neural Networks Learn Language Models? In *Proceedings of the 6th International Conference in Spoken Language Processing (ICSLP'00)*, Beijing (China), 2000.
- [4] M. J. Castro, V. Polvoreda, and F. Prat. Connectionist N-gram Models by Using MLPs. In *Proceedings of the Second Workshop on Natural Language Processing and Neural Networks (NLPNN'01)*, pages 16–22, Tokyo (Japan), November 2001.
- [5] Douglas L. T. Rohde. *A Connectionist Model of Sentence Comprehension and Production*. PhD thesis, Computer Science Department, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2002.
- [6] Masaaki Nagata and Tsuyoshi Morimoto. First steps toward statistical modeling of dialogue to predict the speech act type of the next utterance. *Speech Communication*, 15:193–203, 1994.
- [7] Andreas Stolcke et al. Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Computational Linguistics*, 26(3):339–373, 2000.
- [8] Emilio Sanchis, Isabel Galiano, Fernando García, and Antonio Cano. A Hybrid Approach to the Development of Dialogue Systems directed by Semantics. In *2nd SIGdial Workshop on Discourse and Dialogue*, pages 149–152, Aalborg (Denmark), September 2001.
- [9] S. K. Bennacef et al. A Spoken Language System for Information Retrieval. In *Proceedings of the 3rd International Conference in Spoken Language Processing (ICSLP'94)*, pages 1271–1274, Yokohama (Japan), September 1994.
- [10] H. Bonneau-Maynard and F. Lefèvre. Investigating stochastic speech understanding. In *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU'01)*, 2001.
- [11] Emilio Sanchis, Fernando García, Isabel Galiano, and Encarna Segarra. Applying dialogue constraints to the understanding process in a Dialogue System. In *Proceedings of 5th International Conference on Text, Speech and Dialogue (TSD'02)*, Brno (Czech Republic), 2002.
- [12] María José Castro and Emilio Sanchis. A Simple Connectionist Approach to Language Understanding in a Dialogue System. In *Proceedings of the 8th Conferencia Iberoamericana de Inteligencia Artificial (Iberamia'02)*. Sevilla (Spain), November 2002.
- [13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *PDP: Computational models of cognition and perception*, I, pages 319–362. MIT Press, 1986.
- [14] A. Zell et al. *SNNS: Stuttgart Neural Network Simulator. User Manual, Version 4.2*. Institute for Parallel and Distributed High Performance Systems, University of Stuttgart, Germany, 1998.
- [15] C. M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995.
- [16] L.J. Rodríguez, I. Torres, and A. Varona. Evaluation of sublexical and lexical models of acoustic disfluencies for spontaneous speech recognition in Spanish. In *Proceedings of the 7th European Conference on Speech Communications and Technology (Eurospeech'01)*, Aalborg (Denmark), September 2001.

A Trial Method to Create a Natural Interaction in Interactive Genetic Algorithm

Futoshi SUGIMOTO and Masahide YONEYAMA

Department of Information and Computer Sciences, Toyo University

2100 Kujirai Kawagoe Saitama 350-8585, Japan

f_sugi@eng.toyo.ac.jp; yoneyama@eng.toyo.ac.jp

Abstract. We have been developing a hybrid fitness assignment strategy to realize a natural interaction in IGA. The strategy allows a user to select some individuals and evaluate a grade that shows how the selected individual resembles a target image. In this paper, we will show a method to compose fitness when a user selects two individuals in the hybrid fitness assignment strategy. It is known that better performance is obtained when two individuals are selected in the generations limited with a condition. The condition is equivalent to the actual situation in which it is difficult for a user to select only one individual. The hybrid strategy is useful to realize a more natural interaction in the actual situation.

1. Introduction

In the search system that we have been developing [1], we have a computer search for a target image through an interaction between a user and a computer. IGA (Interactive Genetic Algorithm) is applied in the search. Fitness assignment strategy is an important factor in the IGA. Caldwell and Johnston applied the rating-all method in their work [2]. Smith proposed the picking-some-up method [3]. In our search method, a user first selects one individual resembling a target image most and gives it fitness 1.0. The others are assigned with fitness calculated by the fuzzy reasoning based on the relationship between the selected individual and the others. This method has better performance than the picking-some-up method [4][5]. From the viewpoint of human interface, the latter two methods are superior to the rating-all method, because the user's task is lighter and the processing is quicker. However, we noticed during our experiments that all users weren't happy despite of that lighter burden. It is sometimes difficult for a user to select only one individual, because a couple of individuals may equally resemble a target image, or no individuals may have a close resemblance to a target image. In these situations, a user wants to select a few individuals, and to say how they resemble a target image.

In order to create a more natural interaction considering these requirements, we

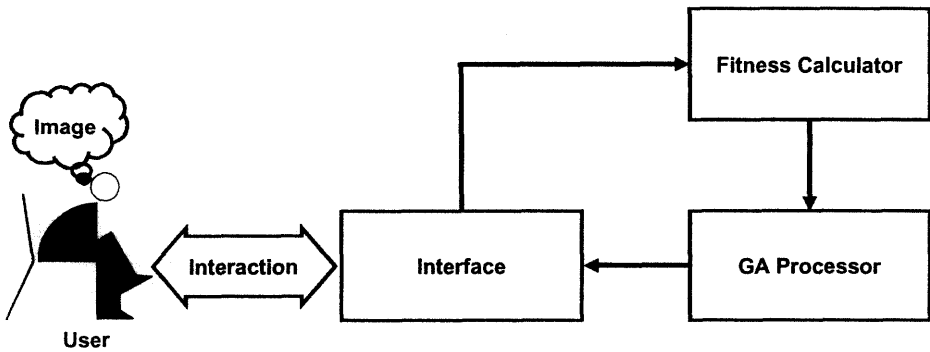


Figure 1. Outline of search system

proposed the hybrid fitness assignment strategy [6]. This strategy allows a user to select some individuals and evaluate a grade that shows how the selected individual resembles a target image. The fitness of all individuals is calculated on the basis of distance between the selected individual and the others. Furthermore, the resembling grade is reflected onto the calculation of fitness with the exponential method. In our previous paper, we limited the number of selected individuals to one. In this paper, we will show a method to compose fitness when a user can select two individuals. From the result of simulation, it is known that the better performance is obtained when two individuals are selected in the generations limited with a condition. The condition is that the resembling grade of the individual nearest to a target is not so large and the deference between the resembling grades of the nearest and the second nearest is very small. This is equivalent to the actual situation mentioned above. The hybrid strategy is useful to realize a more natural interaction in the actual situation.

The organization of the paper is as follows: In section 2, the search system we have been developing is outlined. In section 3, the parameter space that is a searching space and the distance that shows the relation between individuals are defined. In section 4, the hybrid fitness assignment strategy is proposed, and the calculation of fitness and the method to reflect the resembling grade on it are described. In section 5, assumptions and conditions of the simulation in evaluating the strategy are put forth. In section 6, the simulation results are given along with discussing about the results. Finally, in section 7, the conclusions are offered

2. Outline of Search System

In our search system, a computer draws an image that a user has in his or her mind through an interaction between a user and a computer. The user gives a computer some information about the image. Then the computer draws an image based on the information, and asks him whether the image is accurate. If the image is not correct, the user gives the computer more information. Finally, the computer can draw up the image that he or she is thinking

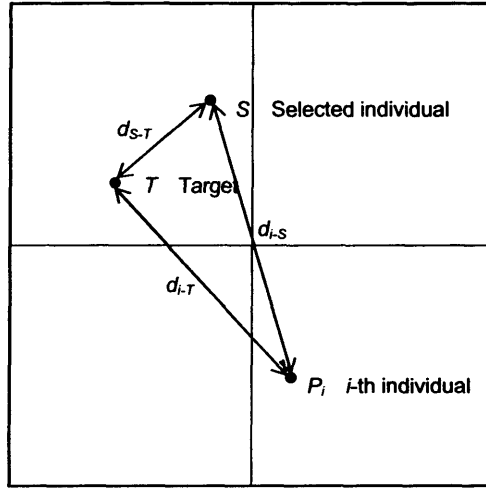


Figure 2. Relation among individuals in parameter space

about. In order to realize the search system, we applied IGA as shown in Figure 1. A computer shows a user some samples of image, which are individuals in GA, through the Interface in each generation. The user inputs the information that is necessary to the adopted fitness assignment strategy. For example, in the rating-all method, the user inputs an evaluation value that indicates how each individual resembles a target image, in the hybrid strategy, the user selects one or a few individuals depending on the situation, and input a resembling grade for each individual. The computer calculates the fitness of all individuals in Fitness Calculator and executes a GA operation in GA Processor, then reproduces individuals of the next generation, and shows them to a user.

3. Parameter Space

All individuals in a generation are corresponded to a point in a space that consists of parameters to draw an image. We will call this space parameter space. The coordinate of an individual i is expressed as follows:

$$P_i = (p_{i1}, p_{i2}, p_{i3}, \dots, p_{in}), \quad p_{min} \leq p_{ij} \leq p_{max} \quad (1)$$

Here, p_{min} and p_{max} are the minimum and maximum values of p_{ij} and n is the number of parameters, or the dimension of the space. The distance between two individuals i and j in parameter space is calculated by the following equation:

$$\text{distance} = \sqrt{\sum_{k=1}^n (p_{ik} - p_{jk})^2} \quad (2)$$

Since each parameter has a different influence on the similarity among individuals, the distance shown in the equation (2) is generally a weighted one as shown in the following equation:

$$\text{distance} = \sqrt{\sum_{k=1}^n w_k (p_{ik} - p_{jk})^2} \quad (3)$$

Here, w_k is a weight of k -th parameter. In the simulation in this paper, however, we do not adopt a concrete subject to search for, so then we use the equation (2), having no weight to calculate the distance.

Although the parameter space is usually multi-dimensional, we illustrate the relation among a target, a selected individual and the others on a two-dimensional plane as shown in Figure 2. Here, d_{S-T} is the distance between a target and a selected individual, d_{i-S} is the distance between i -th individual and a selected one, and d_{i-T} , between i -th individual and a target.

4. Hybrid Strategy

4.1 Calculation of Fitness

In the hybrid strategy, a user can select an optional number of individuals in each generation and input a so-called resembling grade that shows how each selected individual resembles a target image. The selected individual is given maximum fitness 1.0 and the fitness of the others is calculated on the basis of distance between the selected individual and the others in the parameter space. When an individual is selected, i -th individual is given the fitness f_i that is calculated by the following equation:

$$f_i = \left(\frac{d_{\max} - d_{i-S}}{d_{\max}} \right)^a \quad (4)$$

Here, d_{\max} is the maximum distance in the parameter space and a is a constant. When the fitness has a linear proportion to d_{i-S} , the constant a is 1.0, but it is actually set with 3.0 according to the result of trial and error.

4.2 Reflection of Resembling Grade

In order to reflect the resembling grade g on the calculation of fitness, we use the exponential method as shown in the following equation:

$$f_i = \left(\frac{d_{\max} - d_{i-S}}{d_{\max}} \right)^{3.0 \cdot g} \quad (5)$$

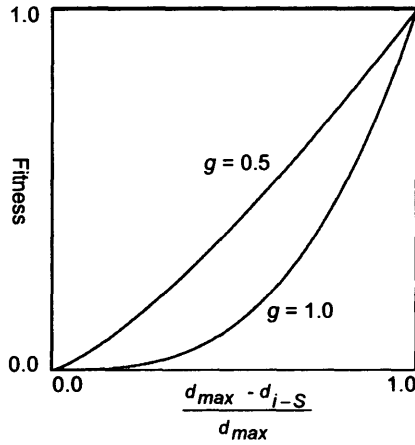


Figure 3. Exponential method

The resembling grade g shows how each selected individual resembles a target image and takes a value between 0.0 and 1.0. Figure 3 illustrates the equation (5), taking $g = 1.0$ and $g = 0.5$. When the resembling grade is larger, the curve sags down and then the influence of a selected individual becomes relatively larger. When the resembling grade is smaller, the curve stretches and reaches the straight line and the influence becomes relatively smaller.

4.3 Composition of Fitness

If a user select two individuals A and B , and their resembling grades are $g(A)$ and $g(B)$ respectively, an individual i has two different fitness, $f_i(A)$ and $f_i(B)$ calculated by the equation (5). Now, we define the composed fitness of the individual i a weighted average with the resembling grades as shown in the following equation:

$$f_i(Com) = \frac{g(A)f_i(A) + g(B)f_i(B)}{g(A) + g(B)} \quad (6)$$

Eventually, we use a normalized value using the maximum of $f_i(Com)$ s, $f_{max}(Com)$ as the fitness of individual i .

$$f_i = \frac{f_i(Com)}{f_{max}(Com)} \quad (7)$$

5. Simulation for Evaluation

We evaluate the hybrid strategy when a user is allowed to select two individuals by simulation using a computer instead of an experiment using subjects. For the simulation, we assume that a user selects an individual nearest to a target when he or she selects only one individual, and selects ones nearest and second nearest when two individuals. The resembling grade g is calculated in the simulation by the following equation:

$$g = \frac{d_{max} - d_{s-T}}{d_{max}} \quad (8)$$

We simulated these three cases in the hybrid strategy: a case of selecting one individual, a case of selecting two individuals through all generations, and a case of selecting two individuals in limited generations satisfying a condition and one individual in the other generations. The condition is that the resembling grade of the individual nearest to a target is less 0.85 and the difference between the resembling grades of the nearest and the second nearest is less 0.03. The condition is equivalent to the situation in which it is difficult for a user to select only one individual, a couple of individuals may equally resemble a target image.

We generated a target and the seven sets of initial individuals by random numbers for the simulation, because the result of GA depends on the initial individuals. The same sets of initial individuals and the same target have been used in all three cases. Since we assume that the parameter space is 12-dimentional and each parameter has 8 possible variations (3 bits), the chromosome length is 36 bits. The parameters of GA operation in the simulation are as follows:

Population size	10
Chromosome length	36
Crossover rate	1.0
Crossover method	Simplex
Simplex crossover rate	0.6
Mutation rate	0.05
Number of elites to survive	1

6. Results of Simulation

The result of a simulation using dataset 1 is shown in Figure 4. y-axis is the distance between the fittest individual and a target, and x-axis is the generation. If two individuals are selected in every generation, the distance between the fittest individual and a target enlarged in the latter generations. On the other hand, if two individuals are selected in generations limited with the condition that the resembling grade of the individual nearest to a target is less 0.85 and the deference between the resembling grades of the nearest and the

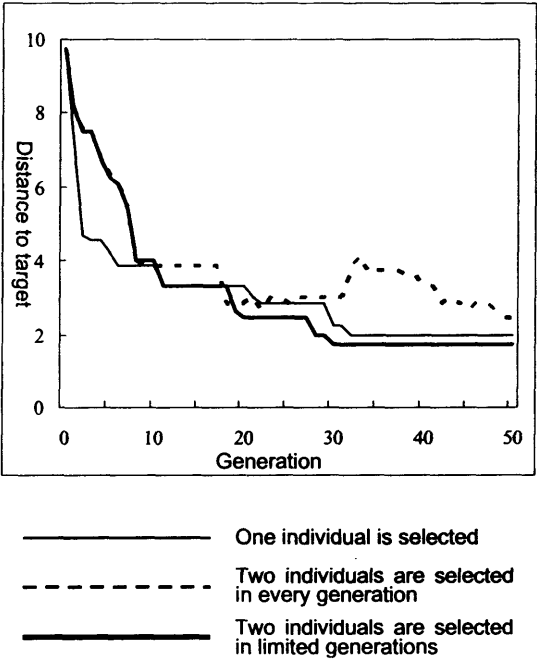


Figure 4. Result of simulation using dataset 1

second nearest is less 0.03, the distance can get smaller than in the case of selecting one individual. The situation satisfying the condition appears in early generations. There is the same result in the other datasets as shown in Figure 5.

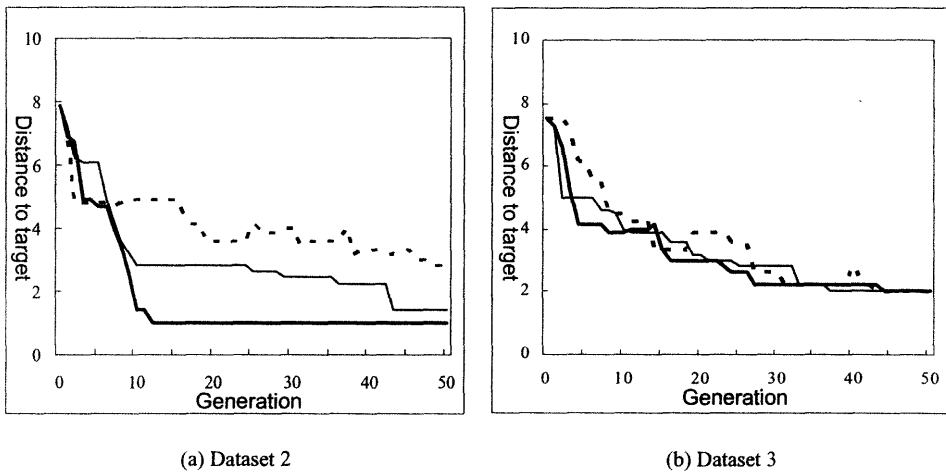
7. Conclusions

Since a sample resembling the image that a user has in his or her mind hardly appears in early generations, the user wants to select more than one sample and to express a resembling grade, such as it resembles a target image a little, or it does not resemble a target image very much. Conversely, a sample resembling a target image easily appears in the latter generations, and then a user can select a sample resembling a target image most. The hybrid strategy can deal with this actual situation. The result of the simulation shows that the hybrid strategy is useful for fitness assignment in IGA, and that by using the strategy we will be able to develop a human interface that realizes a more natural interaction.

In order to improve the interface more, we should allow a user to select not only individuals that resemble a target, but also ones that do not resemble it. Our future work will be determining how to utilize the negative information in IGA.

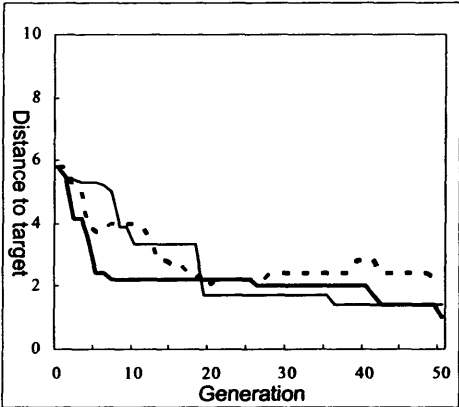
References

- [1] F. Sugimoto, N. Honda and K. Nishio, "A Method of Fitness Assignment in Interactive Genetic Algorithm Using Fuzzy Reasoning Based on Psychological Measure", *Journal of Japanese Society for Artificial Intelligence*, Vol.13, No.5, pp.71-77, 1998 (in Japanese).
- [2] C. Caldwell and V.S. Johnston, "Tracking a Criminal Suspect through Face-Space with a Genetic Algorithm", *Proceedings of ICGA*, pp.416, 1991.
- [3] J.R. Smith, "Designing Bimorphs with an Interactive Genetic Algorithm", *Proceedings of ICGA*, pp.535, 1991.
- [4] F. Sugimoto, "Searching and Drawing a Facial Image in the Mind by Using a Psychometrical Space Model", *Proceedings of International Conference on Advances in Intelligent Systems: Theory and Applications*, pp. 363-365, 2000.
- [5] F. Sugimoto and M. Yoneyama, "Robustness Against Instability of Sensory Judgment in a Human Interface to Draw a Facial Image Using a Psychometrical Space Model", *Proceedings of IEEE International Conference on Multimedia and Expo*, pp. 635-638, 2000.
- [6] F. Sugimoto and M. Yoneyama, "An Evaluation of Hybrid Fitness Assignment Strategy in Interactive Genetic Algorithm", *Proceedings of the 5th Australasia-Japan Joint Workshop on Intelligent & Evolutionary Systems*, pp. 62-69, 2001.

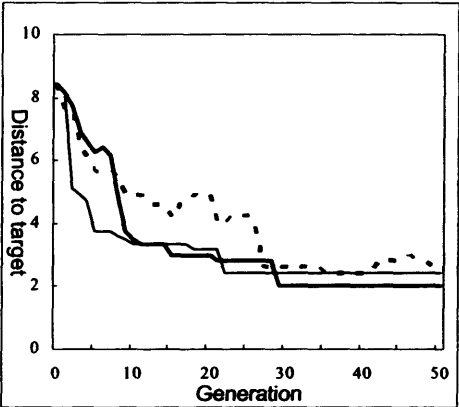


————— One individual is selected
 - - - - - Two individuals are selected in every generation
 ————— Two individuals are selected in limited generations

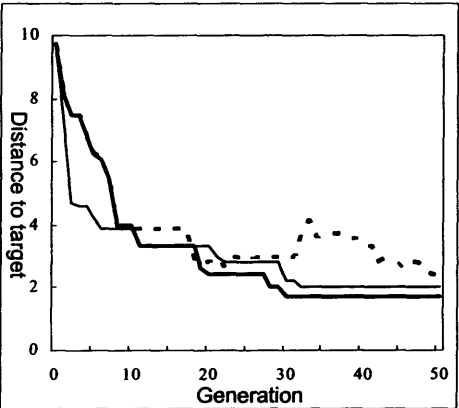
Figure 5. Results of simulation using other datasets



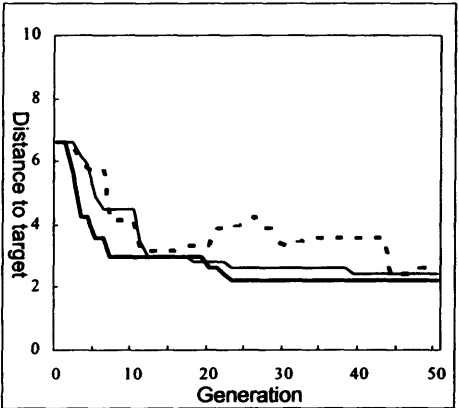
(c) Dataset 4



(d) Dataset 5



(e) Dataset 6



(f) Dataset 7

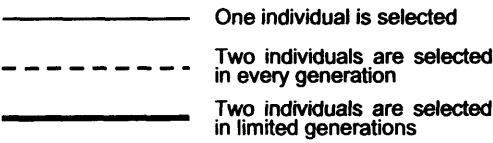


Figure 5. Results of simulation using other datasets (continued)

Section 12

Signal and Image Processing

This page intentionally left blank

Eigenspace-based Face Recognition: A comparative study of different hybrid approaches

Pablo NAVARRETE and Javier RUIZ-DEL-SOLAR
Department of Electrical Engineering, Universidad de Chile.
Email: {pnavarre, jruiroz}@cec.uchile.cl

Abstract. Different eigenspace-based approaches have been proposed for the recognition of faces. They differ mostly in the kind of projection method been used, in the projection algorithm been employed, in the use of simple or differential images before/after projection, and in the similarity matching criterion or classification method employed. Statistical, neural, fuzzy and evolutionary algorithms are used in the implementation of those systems. The aim of this paper is to present an independent, comparative study between some of these hybrid eigenspace-based approaches. This study considers theoretical aspects as well as simulations performed using a small face database (Yale Face Database) and a large face database (FERET).

1. Introduction

Face Recognition is a high dimensional pattern recognition problem. Even low-resolution face images generate huge dimensional feature spaces (20,000 dimensions in the case of a 100x200 pixels face image). In addition to the problems of large computational complexity and memory storage, this high dimensionality makes very difficult to obtain statistical models of the input space using well-defined parametric models. Moreover, this last aspect is further stressed given the fact that only few samples for each class (1-4) are normally available for training the system. Hybrid eigenspace-based approaches have been used in the literature to overcome the mentioned problems. Given that similar troubles are normally found in many biometric applications, we believe that some of the hybrid eigenspace-based methods to be outlined and compared in this work can be applied in the implementation of other biometric systems (signature, fingerprint, iris, etc.).

Among the most successful approaches used in face recognition we can mention eigenspace-based methods, which are mostly derived from the Eigenface-algorithm [14]. These methods project input faces onto a dimensional reduced space where the recognition is carried out, performing a holistic analysis of the faces. Different eigenspace-based methods have been proposed. They differ mostly in the kind of projection/decomposition approach been used (standard-, differential- or kernel-eigenspace), in the projection algorithm been employed, in the use of simple or differential images before/after projection, and in the similarity matching criterion or classification method employed. The aim of this paper is to present an independent, comparative study among some of these different approaches. We believe that to carry out an independent study is important, because comparisons are normally performed using the own implementations of the research groups that have proposed each method (e.g. in FERET contests), which does not consider completely equal working conditions (e.g. exactly the same pre-processing steps).

Very often, more than a comparison between the capabilities of the methods, a contest between the abilities of the research groups is performed. Additionally, not all the possible implementations are considered (e.g. p projection methods with q similarity criteria), but only the ones that some groups have decided to use.

This study considers standard, differential and kernel eigenspace methods. In the case of the standard ones, three different projection algorithms (Principal Component Analysis - PCA, Fisher Linear Discriminant - FLD and Evolutionary Pursuit - EP) and five different similarity matching criteria (Euclidean-, Cosines- and Mahalanobis-distance, SOM-Clustering and Fuzzy Feature Contrast - FFC) were considered. In the case of differential eigenspace methods, two approaches were used, the pre-differential [4] and the post-differential one [9]. In both cases two classification methods, Bayesian and Support Vector Machine - SVM classification, were employed. Finally, regarding kernel eigenspace methods [6], Kernel PCA - KPCA and Kernel Fisher Discriminant - KFD were used.

It is important to point out that in the mentioned approaches a hybrid utilization of statistical, neural, fuzzy and evolutionary algorithms is used. Thus PCA, FLD, KPCA, KFD and the Bayesian classifier correspond to statistical algorithms. SOM and SVM are two well-known neural models. And finally, FFC and EP correspond to fuzzy and evolutionary algorithms, respectively. In this context these algorithms are used in a complementary more than in a competitive way. Requirements as robustness, higher recognition rates, tolerance for imprecision and uncertainty, flexibility, user-friendship and low-cost can be fulfilled by the join use of them.

This comparative study considers theoretical aspects as well as simulations performed using the Yale Face Database, a database with few classes and several images per class, and FERET, a database with many classes and few images per class. It is important to use both kind of databases, because, as it will be shown in this work, some properties of the methods, as for example their generalization ability, changes depending on the number of classes taken under consideration.

The pre-processing aspects of all the approaches (face alignment, illumination invariance, geometrical invariance, etc.) are kept unchanged among their different implementations.

This paper is structured as follows. In section 2 several hybrid approaches for the eigenspace-based recognition of faces are described. In section 3 a comparative study among these hybrid approaches is presented. Finally, some conclusions of this work are given in section 4.

2. Hybrid Approaches for the eigenspace-based Recognition of Faces

Standard eigenspace-based approaches project input faces onto a dimensional reduced space where the recognition is carried out. In 1987 Sirovich and Kirby used PCA in order to obtain a reduced representation of face images [13]. Then, in 1991 Turk and Pentland used the PCA projections as the feature vectors to solve the problem of face recognition, using the Euclidean distance as the similarity function [14]. This system was the first eigenspace-based face recognition approach and, from then on, many eigenspace-based systems have been proposed using different projection methods and similarity functions. A differential eigenspace-based approach that allows the application of statistical analysis in the recognition process, was proposed in 1997 by Pentland and Moghaddam [4]. The main idea is to work with differences between face images, rather than with face images. In this way the recognition problem becomes a two-class problem, because the so-called "differential image" contains information of whether the two subtracted images are of the same class or different classes. In this case the number of training images per class

increases so that statistical information becomes available. The system proposed in [4] used Dual-PCA projections and a Bayesian classifier. Following the same approach, a system using Single-PCA projections and a SVM classifier was proposed in [9].

In the differential case all the face images need to be stored in the database, which slow down the recognition process. This is a serious drawback in practical implementations. To overcome that drawback a so-called post-differential approach was proposed in [9]. Under this approach, differences between reduced face vectors are used instead of differences between face images. This allows decreasing the number of computations and the required storage capacity (only reduced face vectors are stored in the database), without losing the recognition performance of the differential approaches. Both, Bayesian and SVM classifiers were used to implement this approach in [9].

In the Kernel-based approaches KPCA and KFD, non-linear extensions of PCA and FLD respectively, are used as projection algorithms. The main idea behind this approach is to use linear methods applied to high-dimensional mapped vectors instead of the original vectors, and at the same time to avoid the explicit mapping of these vectors by means of the so-called “kernel-trick”. The generalization of linear methods to non-linear ones using kernels, works as follows: if the algorithm to be generalized uses the training vectors only in the form of Euclidean dot-products, then it can be “kernelized”, and all the dot-products like $\mathbf{x}^T \mathbf{y}$ are replaced by a so-called kernel function $K(\mathbf{x}, \mathbf{y})$. If $K(\mathbf{x}, \mathbf{y})$ fulfills the Mercer’s condition, i.e. the operator K is semi-positive definite, then the kernel can be expanded into a series $K(\mathbf{x}, \mathbf{y}) = \sum_i \phi_i(\mathbf{x}) \phi_i(\mathbf{y})$. In this way the kernel represents the Euclidean dot-product on a different space, called feature space F , on which the original vectors are mapped using the eigenfunctions $\phi: \mathcal{R}^N \rightarrow F$. Depending on the kernel function been used, the feature space F can be even of infinite dimension, as the case of Radial Basis Function (RBF) kernel, but we are never working in such space. A unified framework for solving kernel methods was proposed in [10], including the solution of multiclass-KFD, previously unknown. A unified Kernel-based face recognition system was proposed in [6] [7].

Standard-, differential- and kernel-eigenspace approaches for the recognition of faces are described in the following subsections.

2.1. Standard Eigenspace Face Recognition

Fig. 1 shows the block diagram of a generic, standard eigenspace-based face recognition system. Standard eigenspace-based approaches approximate the face vectors (face images) by lower dimensional feature vectors. These approaches consider an off-line phase or training, where the *projection matrix* ($\mathbf{W} \in \mathcal{R}^{N \times m}$), the one that achieve the dimensional reduction, is obtained using all the database face images. In the off-line phase are also calculated the *mean face* ($\bar{\mathbf{x}}$) and the reduced representation of each database image (\mathbf{p}^k). These representations are the ones to be used in the recognition process. The projection methods employed in this work for the reduction of dimensionality are PCA [14], FLD [2] and EP [3]. The similarity matching criteria used for the recognition process are Euclidean-, Cosine- and Mahalanobis-distance, SOM-Clustering, and FFC similarity. All these methods have been analyzed in [8]. Under this standard eigenspace approach a *Rejection System* for unknown faces is implemented by placing a threshold over the similarity measure.

2.2 Differential Eigenspace Face Recognition

Fig. 2 shows the block diagram of a generic, pre-differential eigenspace-based face recognition system. In this approach the whole face images are stored in the database (NT images). Previously the database face images are centered and scaled so that they are

aligned. An input face image is preprocessed and subtracted from each database image. The result of each subtraction is called “differential image” Δ in R^n and it is the key for identification. That because it contains information of whether the two subtracted images are of the same class or different classes. In this way the original problem of NC classes becomes a two-class problem. The so-called differential images (NT) are projected into a reduced space using a given projection method. Thus, each image is transformed into a reduced differential vector δ in R^m . Thereafter the classification of the reduced differential vectors is performed. The result of each classification (S_i) is negative if the subtracted images (each δ) are of different classes and positive in other case. In order to determine the class of the input face image, the reduced vector with maximum positive classification value is chosen, and the class of its initial database image is given as the result of identification. The rejection system acts just when the maximum classification value is negative, i.e. it corresponds to the subtraction of different classes. Dual-PCA and Single-PCA projections have been used as projection methods. The Dual-PCA projections employ two projection matrices: $W_i \in R^{N \times m_i}$ for intra-classes Ω_i (subtractions within equal classes), and $W_e \in R^{N \times m_e}$ for extra-classes Ω_e (subtractions between different classes). Dual-PCA projections are employed together with a Bayesian classifier in order to perform the classification of the differential images [4]. Single-PCA projection employs a single projection matrix $W \in R^{N \times m}$ (standard PCA) that reduces the dimension of the differential face images, and it is used together with a SVM classifier in order to perform the classification [9].

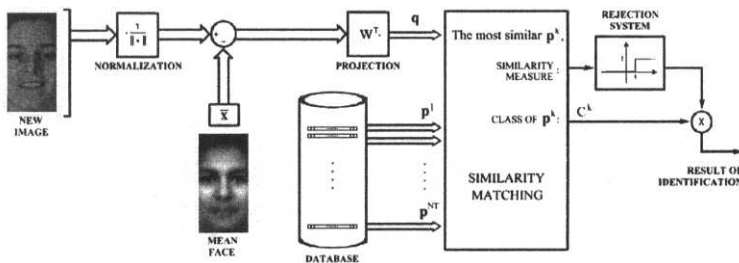


Fig. 1. Block diagram of a generic, standard eigenspace-based face recognition system.

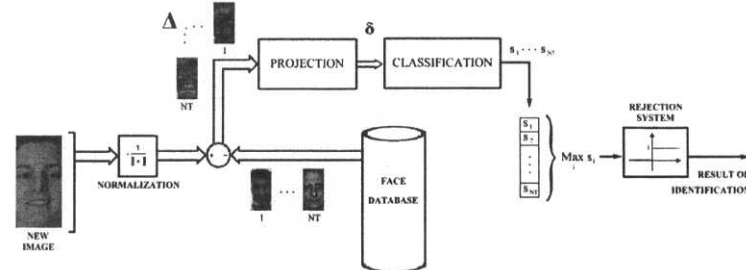


Fig. 2. Block diagram of a generic, pre-differential eigenspace face recognition system.

Fig. 3 shows the block diagram of a generic, post-differential eigenspace-based face recognition system. In this approach only the reduced face images are stored in the database (NT). An input face image is preprocessed and then projected into a reduced space using a given projection method. Thereafter, the new reduced face image is subtracted from each database reduced face image. The result of each subtraction is called “post-differential image” δ in R^m . This vector contains information of whether the two subtracted vectors are of the same class or different classes (intra-classes or extra-classes), and then it works in the same way as a “differential image” once projected on the reduced space. The

classification module performs the classification of the post-differential vectors (NT). The class of the reduced database vector that has the maximum positive classification value gives the class of the initial input face image. If the projection module does not significantly change the topology of the differential-image space, then the pre-differential and post-differential approaches should have very similar recognition rates. The rejection system acts just when the maximum classification value is negative, i.e. it corresponds to the subtraction of different classes. In [9] two different systems that follow this approach were implemented. The first uses Single-PCA projections together with a Bayesian classifier, while the second employs Single-PCA projections together with a SVM classifier.

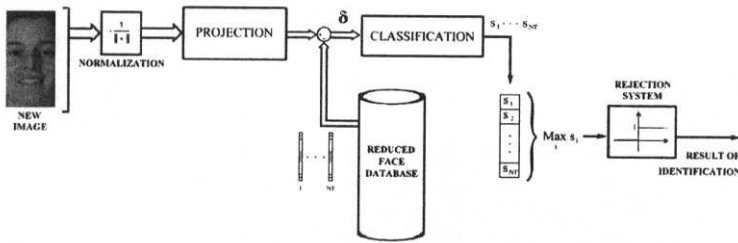


Fig. 3. Block diagram of a generic, post-differential eigenspace face recognition system.

2.3 Kernel Eigenspace Face Recognition

Under this approach the projection of the input face images is carried out in two steps [6]. In the first step, each preprocessed input face image $\mathbf{x} \in \mathcal{R}^N$ (mapped onto the feature space $\phi(\mathbf{x}) \in F$) is projected on the NT support images (mapped in the same feature space), using the kernel function $K: \mathcal{R}^N \times \mathcal{R}^N \rightarrow \mathcal{R}^{NT}$. For doing that in the face database the NT face images needs to be stored. This is due to the fact that kernel machines need all the image vectors in order to reproduce the eigenvectors in the feature space F [10]. In the second step, the parameters of the given kernel machine $\mathbf{A}^T \in \mathcal{M}^{NT \times m}$ (see details in [10]), are applied to the kernel projection vector $\mathbf{k} \in \mathcal{R}^{NT}$, in order to obtain the feature vector $\mathbf{q} \in \mathcal{R}^m$. Afterwards, the *Similarity Matching* module compares the similarity of the reduced representation of the query face vector \mathbf{q} with the reduced vectors \mathbf{p}^k , $\mathbf{p}^k \in \mathcal{R}^m$, that correspond to reduced vectors of the faces in the database. Therefore in the database the reduced vectors needs also to be stored. By using a given criterion of similarity the *Similarity Matching* module determines the most similar vector \mathbf{p}^k in the database. The class of this vector is the result of the recognition process, i.e. the identity of the face. In addition, a *Rejection System* for unknown faces is used if the similarity matching measure is not good enough. The exact computation of $K()$ and \mathbf{A}^T is described in [6][10].

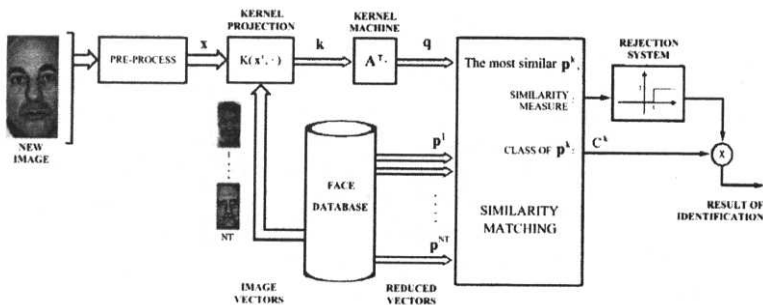


Fig. 4. Block diagram of a generic, kernel-based face recognition system.

3. Comparative Study

The comparative study presented in this section considers theoretical aspects as well as simulations performed using the Yale Face Database, a database with few classes and several images per class, and FERET, a database with many classes and few images per class. It is important to use both kinds of databases, because some properties of the methods, as for example their generalization ability, change depending on the number of classes under consideration. The pre-processing aspects of all the approaches (face alignment, illumination invariance, geometrical invariance, etc.) are kept unchanged among their different implementations.

3.1. Pre-processing

The preprocessing includes: window resizing (scale the face image using fixed proportions, to obtain face images of 100×200), masking (to avoid border pixels not corresponding to the face in the image), illumination gradient compensation (subtract a best-fit brightness plane to compensate heavy shadows caused by extreme lighting angles), histogram equalization (to spread the energy of all intensity values on the image), and normalization (to make all input images of the same energy). All these operations are detailed described in [7].

3.2 Simulations using the Yale Face Image Database

The first simulations were carried out using the Yale University - Face Image Database [15]. We employed 150 images of 15 different classes. First, we pre-processed the images manually by masking them in windows of 100×200 pixels and centering the eyes in the same relative places. In table 1 we show the results of several simulations for the standard eigenspace approaches. For each simulation we used a fixed number of training images, using the same type of images per class, according with the Yale database specification. In order to obtain representative results we take the average of 20 different sets of images for each fixed number of training images. All the images not used for training were used for testing. In tables 2 and 3 we show the results of several simulations using pre-differential and post-differential approaches. We used equal *a priori* probabilities for the Bayes-based methods, $P(\Omega_i) = P(\Omega_e)$, and a penalty for non-separable cases $C=0.01$ in the SVM classification method (see details in [9]). In table 4 we show results using kernel methods. The kernel function was a RBF with $\sigma=0.5$. In KFD the regularization parameter was $\mu=0.05$ (see details in [10]). In tables 1-4 the best results obtained in each experiment, for each approach (standard, pre- and post-differential and kernel), are indicated in bold.

3.3 Simulations using FERET

In order to test the described approach using a large database, we made simulations using the FERET database [12]. We use a target set with 762 images of 254 different classes (3 images per class), and a query set of 254 images (1 image per class). Eyes' location is included in the FERET database for all the images being used. Then the pre-processing considers centering and scaling images so that eyes' position keeps in the same relative place. In table 5 we show the results of simulations for the standard approaches. In this table the SOM-based clustering was not included because in these tests the number of classes (254) is much larger than the number of images per class (3), and the training process is very difficult. In tables 6 and 7 we show the results of simulations using pre-

differential and post-differential approaches. For the Bayesian and SVM classifiers were used the same parameters as before (see 3.2). In table 8 we show results using kernel methods. The kernel function was a RBF with $\sigma=0.5$. In KFD the regularization parameter was $\mu=0.05$ (see details in [10]). In tables 5-8 the best results obtained in each experiment, for each approach (standard, pre- and post-differential and kernel), are indicated in bold.

Tables 1, 2, 3 and 4. Mean recognition rates using the Yale database and different numbers of training images per class, and taking the average of 20 different training sets. The small numbers are standard deviations. All results consider the top 1 match. Whitening is equivalent to use a Mahalanobis distance in a projection space [8].

Table 1. Standard Eigenspace.

projection method	images per class	axes	Euclidean	cos()	SOM	FFC	whitening Euclidean	whitening cos()	whitening SOM	whitening FFC
PCA	6	49	95,7 2,7	95,8 2,7	94,2 2,8	81,8 5,4	83,3 5,9	89,3 4,1	88,8 3,8	81,8 5,4
FLD		14	94,6 2,1	95,2 2,5	93,5 2,4	85,9 5,2	97,2 2,2	97,0 2,5	96,7 3,5	90,8 5,5
EP		21	89,8 4,1	94,3 4,0	92,7 4,2	85,2 3,8	-	-	-	-
PCA	5	42	94,0 2,5	94,1 2,5	92,5 3,1	76,8 10,4	82,2 7,2	87,7 5,6	87,3 5,8	76,8 10,4
FLD		14	94,0 3,2	94,3 2,6	92,8 2,8	87,3 5,8	95,0 3,9	94,2 4,7	93,7 4,5	87,7 6,1
EP		16	93,7 3,2	93,9 2,6	92,1 3,3	88,2 3,6	-	-	-	-
PCA	4	35	93,4 1,9	93,4 2,1	91,6 2,3	78,7 5,5	85,4 4,0	88,0 4,0	79,2 5,3	78,7 5,5
FLD		14	92,9 2,4	93,5 2,4	93,8 2,5	84,7 3,8	94,4 2,1	92,9 3,9	93,1 4,2	85,1 5,7
EP		14	92,3 2,6	92,9 2,5	91,8 2,6	85,3 5,1	-	-	-	-
PCA	3	28	91,9 2,5	92,4 2,2	88,5 2,7	78,6 6,8	84,2 4,0	86,0 4,5	84,0 5,6	78,6 6,8
FLD		14	89,8 4,5	90,9 4,4	85,7 4,6	81,6 5,5	93,0 2,2	92,0 2,8	92,1 2,8	83,9 5,0
EP		14	84,2 3,5	91,2 4,4	88,5 4,4	82,1 5,1	-	-	-	-
PCA	2	21	88,9 5,0	88,9 5,0	79,1 6,1	75,5 6,9	83,4 6,3	85,1 4,0	75,2 6,6	75,5 6,9
FLD		14	88,5 3,4	88,1 4,2	77,2 4,2	79,6 6,1	89,9 4,1	88,1 2,8	86,6 3,4	77,2 7,2
EP		14	77,9 4,8	88,4 5,1	78,5 3,9	78,4 5,1	-	-	-	-

Table 2. Pre-differential Eigenspace.

(i)/(e) indicates intra/extra-classes.

images per class	Axes		Dual-PCA	SVM
	Dual-PCA	SVM		
6	168 (i) / 182 (e)	334	97,5 3,6	96,8 3,4
5	115 (i) / 126 (e)	153	95,5 3,7	94,6 4,1
4	85 (i) / 84 (e)	121	93,2 3,0	92,1 3,4
3	39 (i) / 41 (e)	74	90,4 3,2	91,8 4,4
2	12 (i) / 14 (e)	23	89,5 3,7	90,3 4,4

Table 3. Post-differential Eigenspace.

images per class	axes	Dual-PCA	SVM
6	342	95,6 4,3	97,4 3,6
5	151	92,1 5,3	93,8 4,5
4	115	92,3 4,2	92,6 3,0
3	81	87,9 3,7	91,5 3,8
2	22	86,7 4,8	88,9 5,1

Table 4. Kernel Methods.

projection method	images per class	axes	Euclidean	cos()	SOM	FFC	whitening Euclidean	whitening cos()	whitening SOM	whitening FFC
KPCA	6	89	96,1	96,1	95,1	82,7	92,6	90,7	88,3	82,7
KFD		14	2,7 96,9	2,7 96,8	2,6 95,1	8,9 92,4	4,6 96,3	3,9 93,9	4,3 94,5	8,9 89,8
			3,2	1,2	1,2	4,2	2,4	3,4	4,2	4,8
KPCA	5	74	94,5	94,5	92,3	82,9	88,9	87,7	87,9	82,9
KFD		14	2,5 94,9	2,5 95,4	2,6 91,4	9,9 89,4	7,5 94,5	7,4 92,3	9,2 91,8	9,9 87,6
			4,1	2,8	2,9	5,2	4,1	3,4	5,8	6,1
KPCA	4	59	93,7	93,7	90,6	84,6	89,9	88,1	83,4	84,6
KFD		14	1,9 94,1	1,9 95,7	2,2 91,5	6,2 89,1	4,9 92,9	4,6 91,5	5,8 91,5	6,2 84,3
			3,4	2,4	3,4	4,3	2,6	3,7	4,1	4,8
KPCA	3	44	92,5	92,5	90,5	82,6	90,3	88,1	83,8	82,6
KFD		14	1,9 92,7	1,9 94,0	2,5 90,3	5,5 87,1	3,3 93,0	3,4 91,3	5,4 90,2	5,5 82,3
			2,6	1,8	2,8	5,2	2,4	3,8	3,3	4,8
KPCA	2	29	89,9	89,9	85,2	76,2	90,2	87,7	83,3	76,2
KFD		14	4,3 90,4	4,3 92,3	4,4 88,4	7,8 82,0	3,7 89,1	3,2 87,3	7,5 84,5	7,8 77,5
			2,6	3,6	3,6	4,7	4,1	4,1	5,4	5,7

Tables 5, 6, 7 and 8. Mean recognition rates for standard approaches using FERET. All results consider the top 1 match for recognition. Whitening is equivalent to use a Mahalanobis distance in a projection space [8].

Table 5. Standard Eigenspace.

projection method	images per class	axes	Euclidean	cos()	FFC	whitening Euclidean	whitening cos()	whitening FFC
PCA	3	316	94,1	94,1	87,4	77,6	92,5	87,4
FLD		253	92,5	92,1	91,7	79,9	92,9	90,9
EP		218	92,3	91,8	91,5	-	-	-
PCA	2	252	86,4	86,8	81,5	73,2	85,6	81,5
FLD		253	85,2	85,0	82,9	73,4	79,9	83,1
EP		194	85,7	86,3	83,7	-	-	-

Table 6. Pre-differential Eigenspace.
(i)/(e) indicates intra/extra-classes.

images per class	Axes		Dual-PCA	SVM
	Dual-PCA	SVM		
3	148 (i) / 156 (e)	247	94,2	94,8
2	106 (i) / 128 (e)	192	89,0	88,7

Table 7. Post-differential Eigenspace.

images per class	axes	Dual-PCA	SVM
3	253	92,1	95,2
2	199	87,3	88,5

Table 8. Kernel Methods.

projection method	images per class	axes	Euclidean	cos()	FFC	whitening Euclidean	whitening cos()	whitening FFC
KPCA	3	761	94,5	94,5	85,4	79,5	95,3	85,4
KFD		253	95,3	94,5	95,7	82,7	75,2	60,2
KPCA	2	507	86,6	86,6	83,3	89,6	89,4	83,3
KFD		253	87,8	87,8	88,6	77,2	71,9	62,0

3.4 Results Analysis

By analyzing the Yale-database simulations (tables 1-4), the following can be concluded:

- When using the standard approach, the best results are obtained with the FLD-Withening-Euclidean combination. Using other FLD combinations very similar results are obtained (consider the standard deviation information). FLD uses also less projection axes than the other algorithms.
- Considering the differential approaches, the results obtained using the pre-differential approach are slightly better than the ones obtained with the post-differential one. When the number of images per classes is low (2 or 3) the results for both approaches are very similar (consider the standard deviation information). Dual-PCA and SVM gives similar results in both cases.
- KFD gives better results than KPCA using less projection axes. Very similar results are obtained using Euclidian or Cosines distances.
- The Yale database contains few classes and several images per class (2-6), and in general the best results obtained with each approach are very similar. Differences are seen only when the number of images per classes is low (2 or 3). In this last case, the approaches with better generalization ability, that is the kernel ones, obtains better results. By looking at the standard deviation information it can also be noted that the kernel approaches have a smaller variability in their results. Regarding the number of projection axes employed, the standard approaches use less axes.

By analyzing the FERET-database simulations (tables 5-8), the following can be concluded:

- When using the standard approach, the best results are obtained with PCA. As similarity measure Euclidian or Cosines distances can be used. The number of projection axes used in each approach is of the same order.
- Considering the differential approaches, the results obtained using the pre- and the post-differential approaches are almost identical. In both cases Dual-PCA and SVM gives similar results.
- Results obtained with KFD and KPCA are very similar. However, KFD gives better results when the number of images per class is 3. On the other hand, when the number of images per class is 2, the best results are obtained with KPCA. KFD uses less projection axes than KPCA.
- The FERET database contains many classes and few images per class (2-3), for this reason the best results are obtained when using the approaches with better generalization capabilities, i.e. the kernel ones.

Other information that should also be considered:

- Post-differential approaches are 2 to 5 times faster than the pre-differential ones. Taking in to account their similar recognition rates, post-differential approaches are the better differential alternative.
- Kernel-projections are 2 to 3 times slower than linear projections due to the use of the support images (all the database images). Another drawback of these methods is that the kernel parameters adjustment is very difficult and data dependant.

4. Conclusions

The aim of this paper was to present an independent, comparative study among different eigenspace-based approaches. The study considered standard, differential and

kernel eigenspace methods. In the case of the standard ones, three different projection algorithms (PCA, FLD and EP) and five different similarity matching criteria (Euclidean-, Cosines- and Mahalanobis-distance, SOM-Clustering and FFC-similarity) were considered. In the case of the differential methods, two approaches were used, the pre-differential and the post-differential. In both cases Bayesian and SVM classification were employed. Finally, regarding kernel methods, KPCA and KFD were used.

Simulations were performed using the Yale Face Database, a database with few classes and several images per class, and FERET, a database with many classes and few images per class. By looking at the obtained results it can be concluded:

- Considering recognition rates, generalization ability and processing time, the best results were obtained with the post-differential approach, using either Dual-PCA or SVM.
- In the specific case of the Yale Face Database, where the requirements are not so high, using any of the approaches gives similar results.
- The drawbacks of the kernel methods are their low processing speed and the difficulty to adjust the kernel parameters. The first drawback could be overcome by using a kind of support vectors in the KPCA and KFD algorithms. For sure this is a problem to be tackled by kernel-machine researchers in the next few years.

As future work we will like to extend our study by considering other kernel approaches and algorithms, as for example Independent Component Analysis – ICA and Kernel-ICA.

Acknowledgements

This research was funded by the joint "Program of Scientific Cooperation" of CONICYT (Chile) and BMBF (Germany) under the project "Face Recognition and Signature Verification using Soft-Computing" and by Millennium Nucleus Center for Web Research, Grant P01-029-F, Mideplan, Chile. Portions of the research in this paper use the FERET database of facial images collected under the FERET program.

References

- [1] C.J.C. Burges, "A tutorial on support vector machines for pattern recognition", *Data Mining and Knowledge Discovery*, 2(2), pp. 121–167, 1998.
- [2] R.A. Fisher, "The Use of Multiple Measures in Taxonomic Problems", *Ann. Eugenics*, vol. 7, pp. 179–188, 1936.
- [3] C. Liu and H. Wechsler, "Evolutionary Pursuit and Its Application to Face Recognition", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 570–582, June 2000.
- [4] A. Pentland and B. Moghaddam, "Probabilistic Visual Learning for Object Representation", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 696–710, July 1997.
- [5] R.O. Duda, P.E. Hart and D.G. Stork, "Pattern Classification", Second Edition, 2001.
- [6] P. Navarrete and J. Ruiz-del-Solar, "Kernel-based Face Recognition by a Reformulation of Kernel Machines", *Proc. of the 7th Online World Conf. on Soft Computing in Industrial Applications*, WSC 7, September 2002.
- [7] P. Navarrete, "Face Recognition and Face Retrieval using Statistical Learning Techniques", Thesis for the title of Electrical Engineer, Department of Electrical Engineering, Universidad de Chile, June 2002 (in Spanish).
- [8] P. Navarrete and J. Ruiz-del-Solar, "Comparative study between different Eigenspace-based approaches for Face Recognition", *Lecture Notes in Artificial Intelligence* 2275, AFSS 2002, Springer, 178–184.
- [9] J. Ruiz-del-Solar and P. Navarrete, "Toward a Generalized Eigenspace-based Face Recognition System", *Lecture Notes in Computer Science* 2396 (Structural, Syntactic, and Statistical Pattern Recognition), Springer, 662–671.
- [10] P. Navarrete and J. Ruiz-del-Solar, "On the Generalization of Kernel Machines", *Lecture Notes in Computer Science* 2388 (Pattern Recognition with Support Vector Machines - SVM 2002), Springer, 24–39.
- [11] C. Cortes and V. Vapnik, "Support Vector Networks", *Machine Learning*, 20, pp. 273–297, 1995.
- [12] P. J. Phillips, H. Wechsler, J. Huang and P. Rauss, "The FERET database and evaluation procedure for face recognition algorithms", *Image and Vision Computing J.*, Vol. 16, no. 5, 295–306, 1998.
- [13] L. Sirovich and M. Kirby, "A low-dimensional procedure for the characterization of human faces", *J. Opt. Soc. Amer. A*, vol. 4, no. 3, pp. 519–524, 1987.
- [14] M. Turk and A. Pentland, "Eigenfaces for Recognition", *J. Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.
- [15] Yale University Face Image Database, publicly available for non-commercial use, <http://cvc.yale.edu/projects/yalefaces/yalefaces.html>.

Pattern Recognition with Ultrasonic Sensor using Classification Methods

Naïma AIT OUFROUKH, Etienne COLLE

CEMIF - Complex Systems Group, 40, Rue du Pelvoux, Evry, France

Abstract. This paper describes a binaural ultrasonic sensor for mobile robot recognition of simple objects with non-parametric methods (K nearest neighbours and a neural network). These methods identify and then exploit echo features defined as characteristics of the simple objects. The features selection is also studied and the reduction of parameters is obtained with several methods: Sequential Backward, Forward Selection and Branch and bound. The parameters correlation is verified by correlation circle given by the principals component analysis. The result of feature selection and classification are presented.

1. Introduction

Ultrasonic ranging sensors have been widely used for determining the proximity of objects and have been employed for robot localization and map building [1,2]. In our study, sonar is used to recognize simple objects of indoor environment. The limitation of sonar is due to a poor lateral resolution because of the relatively wide directivity of the transducer [1,4]. The only used information being the time of flight (TOF). Furthermore, the knowledge of the time of flight is insufficient when it is about pattern recognition. It is necessary to enrich this information with the measure of parameters characterising in a reliable way the signature of the received signal. The problem is to determine the most discriminating parameters between the environment objects. In literature, we found the differential TOF models of targets [8], combining amplitude, energy, and duration of the echo signals along with TOF information [9], or the echo amplitude [1,2]. However, a major problem with using the amplitude information of sonar signal is that the amplitude is very sensitive to environmental conditions. For this reason, and also because the standard electronics which saturates following some measurement conditions, amplitude information is seldom used only. In the present paper, we employ all the information that we can extract from the sonar signal (Amplitude, Frequency, Length, etc...). The non-parametric methods (K nearest neighbour and neural network) are used for processing meaningful features so as to reliably handle the target classification problem. The Sequential Backward, Forward selection and branch and bound approaches are used to reduce the number of discriminating features to classify the echoes.

2. Model of the environment

The two major components of the ultrasonic ranging system are the transducer and the drive electronics. In operation, a pulse of ultrasonic sound is transmitted towards a target and the resulting echo is detected. The elapsed time between the start of the transmit pulse

and the reception of the echo pulse is measured. Since the speed of the sound in air is known, the system can convert the elapsed time into a distance measurement.

We concentrate on environments composed of specular surfaces [9]. The target primitives modelled in this study are *plane*, *corner*, *edge* and *cylinder*, of different dimension. The corner and edge are being defined like two perpendicular planes.

3. The bi-aural sensor system

In our system, two identical acoustic transducers (Polaroid) employed to improve the angular resolution [10]. Each transducer can operate both as transmitter and receiver (Figure 1).

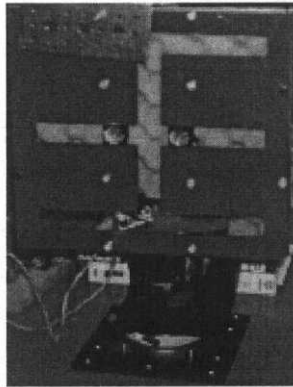


Figure 1: Bi-aural system

4. Classification

The choice of components of the entry vector is one of the major elements for a good classification. The originality of our approach is to have selected components of the vector according to the objective to reach and not a priori. Before defining the classifier it has been done a parameter selection.

4.1. Dimensionality Reduction

There are two main reasons to keep the dimensionality of the pattern representation (i.e., the number of features) as small as possible: measurement cost and classification accuracy. A limited yet salient feature set simplifies both the pattern representation and the classifiers that are built on the selected representation. Consequently, the resulting classifier will be faster and will use less memory. Moreover, as stated earlier, a small number of features can alleviate the curse of dimensionality when the number of training samples is limited. On the other hand, a reduction of the number of features may lead to a loss in the discrimination power and thereby decreases the accuracy of the resulting recognition system. The classification depends strongly on the features selected to represent the sample.

Feature extraction is achieved by an heuristic method. Good feature selection method is an essential step in a classification system [3], which permits to reduce the space of class representation. We use the following criteria :

$$J = \text{trace}(\hat{\Sigma}_w^{-1} \hat{\Sigma}_b^{-1}) \quad (1)$$

Where $\hat{\Sigma}_w$ is the estimated intra group covariance matrix, which characterizes the dispersion of points in a class. $\hat{\Sigma}_b$ is the estimated inter group covariance matrix which characterizes the dispersion of classes between them.

We made comparison of two feature selection algorithms, with the rate of classification. Forward and Backward sequential selection (FSS, BSS) [3] are the most common sequential search algorithms. FSS begins with zero features, evaluates all subsets with exactly one feature and selects the one with largest criteria. BSS instead begins with all features and repeatedly removes a feature whose removal causes the least decrease of criteria. At the k step, we add or remove the feature ξ_j as:

$$J(\Xi_k \pm \xi_j) \geq J(\Xi_k \pm \xi_i) \quad i=1, d-k \quad i \neq j \quad (2)$$

Where J is the criteria (1), Ξ_k is the whole set of features minus k features. The remaining parameters are most discriminative.

After a rough selection of parameters we apply the branch and bound method (BAB) to refine the result. This method examines all feature subsets. It will always find an optimal solution but with cost of computational time.

The selected parameters contain the main features of the signal received by each sensor (Figure 2). We can regroup these parameters in four categories: the maximal amplitude, the shape, the energy and the difference of viewpoints between these two signals. The initial set of parameters is composed of :

- 20 components of transform of Fourier, length and the amplitude of the received signal,
- the distance between targets and each receiver, and the time of maximum
- The difference between the two measured time of flight by each receiver,
- The maximal amplitude difference,
- The length difference.

This gives an initial set of 47 parameters.

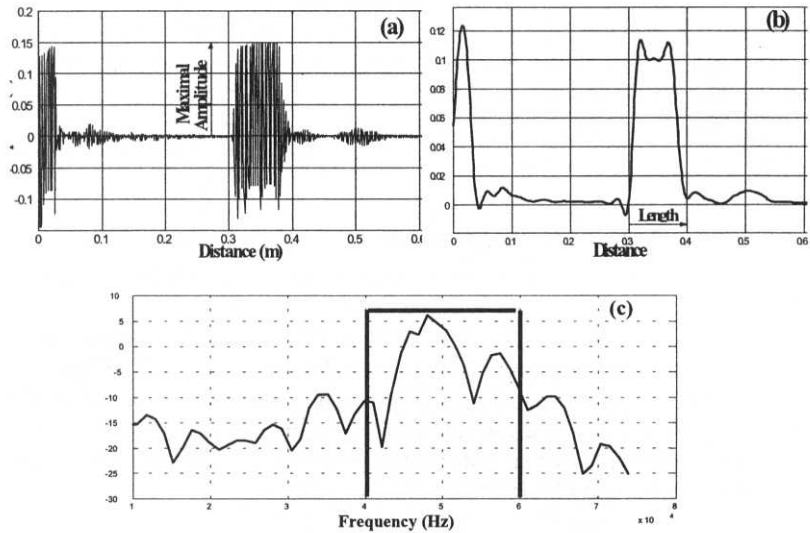


Figure 2: Feature extraction

The figure 2.a shows a real received signal (Amplitude versus Distance). The figure 2.b shows the envelope of the echo (Amplitude versus Distance) and the figure 2.c fast Fourier transformed of the received signal.

4.2. Classification methods

The two classification methods used to discriminate between different objects are neural network and K nearest neighbours.

The first method is Levenberg-Marquardt neural net classifier (NNet). The choice of neural network architecture in terms of input number and output number is conducted by the characteristics of the application. In our case, networks are composed of three hidden layers with a non linear sigmoid activation contrary to input and output layers which use linear activation functions. The best architecture is chosen after comparing the generalization error for different initializations and hidden layer sizes. Data set is divided into three equal's parts: a learning data set, a test data set and a generalization data set. A softmax function [7] evaluates a posteriori probability of neural classification:

$$\Pr(k/x)=\frac{\exp(y_k)}{\sum_j^k \exp(y_j)} \tag{3}$$

y_j is the j output of the neural classifier

The second method is the nearest neighbour classifier. It does not require any user-specified parameters except the distance metric used to find the nearest neighbour, but Euclidean distance (d) is commonly used. The nearest neighbour is defined as:

$$\hat{w}(x) = w(x) \quad \text{si} \quad d(x, x') = \text{Min}_{j=1, n} d(x, x_j) \quad (4)$$

$\hat{w}(x)$ is the class of valued affectation of x

x' is the near neighbour of x .

5. Results

5.1 Experimentation

5.1.1 Data base

Two bases of measures have been done, 2000 examples for the training and another for the generalization. Each one is composed of the four classes, plane, corner, cylinder and edge. The number of examples depends on the dimension of object. For the plane, we use two planes of 16cm and 122cm width, the corner 25cm and 48cm width, the edge 20cm and 43cm width and for cylinder, we use three with different diameter: 16mm, 4cm and 20cm. The distance between the center of sensor and soil is 37cm. The sensor target distance varies between 0.20m to 2.30m, with an increment of 20cm. The orientation of the sensor varies between $\pm 20^\circ$ with an increment of 5° (i.e. for the cylinder 16mm the angle of orientation varies between $\pm 8^\circ$). The distance between the two transducers is $d = 8\text{cm}$, because the maximal distance between the two transducers is 10cm [5].

5.1.2 Determination of discriminating parameter

Choosing the feature vector dimension is usually a compromise between classification performance and computational time. We compare the FSS and BSS methods, according to the given rate classification by quadratic discriminant analysis (QDA) [6], this rule is based on the normal distribution:

$$f_k(X) = (2\pi)^{-p/2} |\Sigma_k|^{-1/2} e^{-\frac{1}{2}(X - \mu_k)^T \Sigma_k^{-1} (X - \mu_k)} \quad (5)$$

Where μ_k and Σ_k are the class k ($1 \leq k \leq K$) population mean vector and covariance matrix. After comparison, we find that BSS gives the best result in term of classification.

In this paper we reduce the number of parameter from 47 to 15 parameters with the backward sequential method, to refine the final set, we apply the BAB method. We could not use them for more than 15 parameters because of the importance of the computation time.

Table 1: Features selection

	Backward				Optimal	
Method	QDA				QDA	
Parameter number	47	40	20	15	10	4
Plane %	93	94	95	91	90	91
Corner %	74	75	90	91	88	71
Edge %	74	73	84	83	70	72
Cylinder %	80	79	82	78	74	72

The table 1 shows the different rate of classification with different number of parameter using QDA.

We reduce the dimension of patterns to 4 parameters that are:

- Two frequencies : 50kHz of transmitter and receiver
- Signal amplitude for the second receiver
- Difference between the two time of flight,

To determine the most uncorrelated parameters we use the principal component analysis (PCA)[3]. The parameter is considered meaningful when it is close to the side of the circle.

To verify the correlation between these features, we present the correlation circle (Figure 3).

We remark that the four parameters (P1⇒50kHz, P2 ⇒ 50kHz, P3 ⇒ Maximal Amplitude and P4 ⇒ Difference of TOF) are uncorrelated and meaningful.

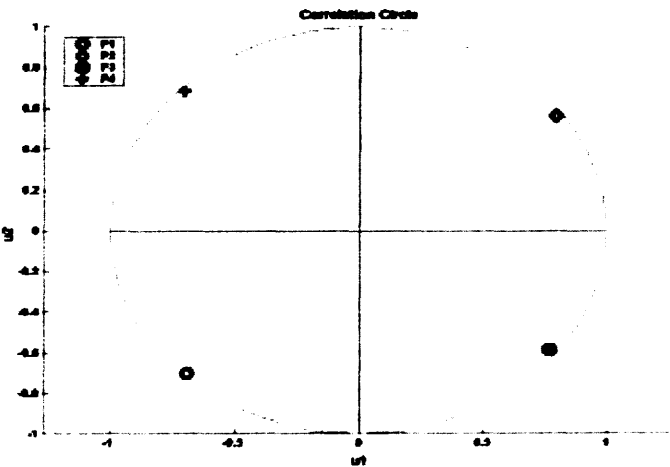


Figure 3: Correlation circle of parameters

(u_1, u_2) are principal axes of PCA.

5.2 Classification results

For the K nearest neighbours, the choice of K is heuristic. In [3] we can take it equal to \sqrt{n} or relatively weak in relation to n : n is the number of parameters. In our case $K=3$.

Table 2 show the confusion matrix for different objects with NNet and 3NN methods.

Table 2: Rate of classification with NNet and 3NN

		Plane	Corner	Edge	Cylinder
3NN	Plane	91%	20%	9%	21%
	Corner	3%	64%	8%	0%
	Edge	0%	16%	72%	14%
	Cylinder	6%	0%	11%	66%
NNet	Plane	98%	9%	6%	13%
	Corner	1%	86%	9%	1%
	Edge	0%	5%	80%	7%
	Cylinder	1%	0%	5%	79%

The neural network method offers a best rate of classification that the 3 nearest Neighbors, with four parameters. For NNet, the results show a small confusion between the different objects; cylinders of diameter 20cm are confused to planes (13%).

6. Conclusion

In the paper we present two methods to classify 2D indoor environment based on ultrasonic measures. The classification method requires only two ultrasonic sensors and four discriminating parameters. It appears that time of flight, amplitude and resonance frequency are able to separate the different object classes despite some drawbacks of US sensors: saturation, narrow bandwidth (45kHz to 55kHz) and wide beam aperture. The information included in the echo is sufficient to classify the environment in 4 classes (edge, corner, plan and cylinder) with good performances. It is necessary to test this approach on complex objects. For future works it is important to refine the evaluation of the method performances in terms of quality but also of processing time.

References

- [1] B.Barshan, R.Kuc, Differentiating sonar reflections from corners and planes by employing an intelligent sensor, IEEE Trans.Pattern Anal. Machine Intell, June 1990, pp. 560-569.
- [2] L.Kleeman et al, Mobile Robot Sonar For Target Localization And Classification ,IJRR, 1995, pp. 295-318.
- [3] B. Dubuisson, Diagnostic et reconnaissance des formes, Editions Hermes, Septembre 1990.
- [4] B. Barshan et al, Neural Network-Based Target Differentiation Using Sonar for Robotics Applications IEEE Trans. On Robotics and Automation, 2000, pp. 435-442

- [5] N. Ait Oufroukh et al, Distinction between objects with ultrasonic bi-aural system and only amplitude, 8th IEEE on Emerging Technologies and factory automation (ETFA), Antibes, 2001, pp. 758-760.
- [6] J.H.Friedman, Regularized Discriminant Analysis , SLAC-PUB-4389 (REV), July 1988.
- [7] C. Bishop, Neural Networks for Pattern Recognition, Clarendon Press, Oxford, 1995.
- [8] Ö.Bozma and R.Kuc, Building a sonar map in a specular environment using a single mobile sensor, IEEE Trans. Pattern Anal. Machine Intell, 1991, pp. 1260-1269.
- [9] N. Ait Oufroukh et al, Ultrasonic multi-transducer processing for pattern recognition, The first IEEE International Conference on Sensors, Orlando, 2002.

A Color Pattern Recognition Problem based on the Multiple Classes Random Neural Network Model

Jose Aguilar
CEMISID. Dpto. de Computación
Facultad de Ingeniería
Universidad de los Andes
Av. Tulio Febres. Mérida-Venezuela
Email: aguilar@ing.ula.ve
Phone (58.74) 402914
Fax (58.74) 402872

Abstract. The purpose of this paper is to describe the use of the multiple classes random neural network model to recognize patterns having different colors. We propose a learning algorithm for the recognition of color patterns based upon the non-linear equations of the multiple classes random neural network model using gradient descent of a quadratic error function. In addition, we propose a progressive retrieval process with adaptive threshold value. The experimental evaluation shows that our approach provides good results.

1. Introduction

Humans use color, shape and texture to understand and recollect the contents of a pattern. Therefore, it is natural to use features based on these attributes for pattern recognition [8, 9, 16, 17]. In [15] is demonstrated the effectiveness of using simple color features for pattern recognition. Colombo et al. described a system for pictorial content representation and recognition based on color distribution features [8]. They describe the distribution of chromatic content in a pattern through a set of color histograms and a pattern matching strategy using these sets. Recently, Mojsilovic et al. determined the basic categories (vocabulary) used by humans in judging similarity of color patterns, their relative importance and relationships, as well as the hierarchy of rules (grammar) [17].

The problem addressed in this paper concerns the proposition of a color pattern recognition approach composed by a learning algorithm and a retrieval procedure for the multiple classes RNN. We shall use each class to model a color. We present a backpropagation type learning algorithm for the recurrent multiple classes RNN model using gradient descent of a quadratic error function when a set of input-output pairs is presented to the network. Our model is defined for nC parameters for the whole network, where C is the number of primary colors, n is the number of pixels of the image, and each neuron is used to obtain the color value of each pixel in the bit map plane. The primary colors create different colors according to the RGB model. Thus, our learning algorithm requires the solution of a system of nC non-linear equations each time the n -neurons network learns a new input-output pair (n -pixels image with C primary colors). In addition, we propose a progressive retrieval process with adaptive threshold value.

The Random Neural Network (RNN) has been proposed by Gelenbe in 1989 [11, 12, 13]. This model does not use a dynamic equation, but uses a scheme of interaction among neurons. It calculates the probability of activation of the neurons in the network. Signals in this model take the form of impulses that mimic what is presently known of inter-neural

signals in biophysical neural networks. The RNN has been used to solve optimization [1, 2, 4] and pattern recognition problems [3, 5, 7]. Gelenbe has considered learning algorithm for recurrent random neural network model [14]. We have proposed modifications of this algorithm for combinatorial optimization problems [4] and an evolutionary learning for combinatorial optimization and recognition problems [1, 5]. Fourneau et al. have proposed an extension of the RNN, called multiple classes random neural network model [10].

This work is organized as follows, in section 2 we present the multiple classes RNN. Section 3 presents our recognition algorithm (learning and retrieval processes) for multiple classes RNN. In section 4, we present applications. Remarks concerning future work and conclusions are provided in section 5.

2. The Multiple Classes Random Network Model

The neural network is composed of n neurons and receives exogenous positive (excitatory) and negative (inhibitory) signals as well as endogenous signals exchanged by the neurons. As in the classical model 1989 [11, 12, 13], excitatory and inhibitory signals are sent by neurons when they fire, to other neurons in the network or to outside world. In this model, positive signals may belong to several classes and the potential at a neuron is represented by the vector $\underline{K}_i = (K_{i1}, \dots, K_{iC})$, where K_{ic} is the value of the "class c potential" of neuron i , or its "excitation level in terms of class c signals", and negative signals only belong to a single class. The total potential of neuron i is $K_i = \sum_{c=1}^C K_{ic}$. The arrival of an excitatory signal of some class increases the corresponding potential of a neuron by 1, while an inhibitory signal's arrival decreases it by 1. That is, when a positive signal of class c arrives at a neuron, it merely increases K_{ic} by 1, and when a negative signals arrives at it, if $K_i > 0$, the potential is reduced by 1, and the class of the potential to be reduced is chosen randomly with probability K_{ic}/K_i for any $c=1, \dots, C$. A negative signal arriving at a neuron whose potential is zero has no effect on its potential.

Exogenous positive signals of class c arrive at neuron i in a Poisson stream of rate $\Lambda(i, c)$, while exogenous negative signals arrive at it according to a Poisson process of rate $\lambda(i)$. A neuron is excited if its potential is positive. It then fires, at exponentially distributed intervals, sends excitatory signals of different classes, or inhibitory signals, to other neurons or to the outside of the network. That is, the neuron i can fire when its potential is positive ($K_i > 0$). The neuron i sends excitatory signals of class c at rate $r(i, c) > 0$, with probability K_{ic}/K_i . When the neuron fires at rate $r(i, c)$, deletes by 1 its class c potential and sends to neuron j a class ϕ positive signal with probability $p^+(i, c; j, \phi)$, or a negative signal with probability $p^-(i, c; j)$. On the other hand, the probability that the deleted signal is sent out of the network, or that it is "lost", is $d(i, c)$. Clearly we shall have:

$$\sum_j (p^+(i, c; j, \phi) + \sum_j p^-(i, c; j) + d(i, c)) = 1 \quad \text{for } \forall i=1, n \text{ and } c=1, C,$$

Let $K(t)$ be the vector representing the state of the neural network at time t and $K = (K_1, \dots, K_n)$ be a particular value of the vector. We shall denote by $p(K, t) = \Pr[K(t)=K]$ the probability distribution of its state. The main property of this model is the excitation probability of the "class ϕ potential" of neuron j , $q(j, \phi)$, with $0 < q(j, \phi) < 1$, which satisfy a non-linear equation [10]:

$$q(j, \phi) = \lambda^+(j, \phi) / (r(j, \phi) + \lambda^-(j)) \quad (1)$$

where, $\lambda^+(j, \phi) = \sum_{i,c} q(i, c) r(i, c) p^+(i, c; j, \phi) + \Lambda(j, \phi)$

$$\lambda^-(j) = \sum_{(i,c)} q(i, c) r(i, c) p^-(i, c; j) + \lambda(j)$$

Thus, $p(K, t)$, the stationary probability distribution of network state, satisfies

$$p(K, t) = \prod_{i=1}^n \prod_{c=1}^C [1 - q(i, c)] q(i, c)^{K_{ic}}$$

The synaptic weights for positive ($w^+(i, c; j, \varphi)$) and negative ($w^-(i, c; j)$) signals are defined as:

$$w^+(i, c; j, \varphi) = r(i, c) p^+(i, c; j, \varphi)$$

$$w^-(i, c; j) = r(i, c) p^-(i, c; j)$$

and, if $d(i, c)=0$, the fire rate $r(i, c)$ will be

$$r(i, c) = [\sum_{j, \varphi} w^+(i, c; j, \varphi) + \sum_j w^-(i, c; j)]$$

3. Color Pattern Recognition Algorithm on the Multiple Classes Random Neural Network Model

We now show how the multiple classes RNN can be used to solve the Color Pattern Recognition problem. The recognition procedure is based on an associative memory technique [3, 5]. In our approach, a "signal class" represents obviously each color. To design such a memory, we have used a single-layer RNN of n fully interconnected neurons. For every neuron i the probability that emitting signals depart from the network is $d(i, c)=0$. We suppose a pattern composes by n points (m, k) in the plane (for $m=1, \dots, J$ and $k=1, \dots, K$). We associate a neuron $N(i)$ to each point (m, k) in the plane (for $i=1, \dots, n$; $m=1, \dots, J$ and $k=1, \dots, K$). The state of $N(i)$ can be interpreted as the color intensity value of the pixel (m, k) . That is, each pixel is represented by a neuron. In another hand, we suppose three classes to represent the primary colors (red, green, and blue) according to the RGB model. This model allows create different colors with the combination of different intensities of the primary colors. For example, to represent a pixel with red color the neuron value is $(1, 0, 0)$, the black color is $(1, 1, 1)$, the pink color is $(0.5, 0, 0)$, etc. We suppose values equal to 0, 0.5 and 1 for each class on every neuron. In this way, we can represent geometric figures with different combinations of colors. We have used this model because it agrees better with human chromatic perception [8], but our approach can use another model like this one to represent the colors of a given pattern. In order to select a color, we need to take in account that each primary color will be a class, whose increases the complexity of the system. The parameters of the neural network will be chosen as follows:

$$a) \quad p^+(j, \varphi; i, c) = p^+(i, c; j, \varphi) \quad p^-(i, c; j) = p^-(j, c; i) \quad \text{for any } i, j=1, \dots, n \text{ and } c, \varphi=1, \dots, C.$$

$$b) \quad \Lambda(i, c) = \Lambda_{ic} \text{ and } \lambda(i)=0, \text{ where } \Lambda_{ic} \text{ is a constant for the class } c \text{ of the neuron } i.$$

3.1 Learning Algorithm

Now, we search to define a learning algorithm for the multiple classes RNN model. We propose a gradient descent algorithm for choosing the set of network parameters $w^+(j, z; i, c)$ and $w^-(j, z; i)$ in order to learn a given set of m input-output pairs (X, Y) where the set of successive inputs is denoted by:

$X = \{X_1, \dots, X_m\}$ where, $X_k = \{X_k(1,1), \dots, X_k(n, C)\}$, and $X_k(i, c)$ is the c^{th} class on the neuron i for the patron k
 $X_k(i, c) = \{\Lambda_k(i, c), \lambda_k(i)\}$

and the successive desired outputs are the vector

$Y = \{Y_1, \dots, Y_m\}$ where, $Y_k = \{Y_k(1,1), \dots, Y_k(n, C)\}$, and $Y_k(1,1) = \{0, 0.5, 1\}$

The values $\Lambda_k(i, c)$ and $\lambda_k(i)$ provide the network stability. Particularly, in our model $\Lambda_k(i, c)$ and $\lambda_k(i)$ are initialized as have been defined previously. Normally, arrival rates of exogenous signals are chosen as follows:

$$Y_k(i, c) > 0 \Rightarrow X_k(i, c) = (\Lambda_k(i, c), \lambda_k(i)) = (Kc, 0)$$

$$Y_k(i, c) = 0 \Rightarrow (\Lambda_k(i, c), \lambda_k(i)) = (0, 0)$$

The network approximates the set of desired output vectors in a manner that minimizes a cost function E_k :

$$E_k = 1/2 \sum_{i=1}^n \sum_{c=1}^C [q_k(i, c) - Y_k(i, c)]^2$$

The rule to update the weights may be written as:

$$w_k^+(u, p; v, z) = w_{k-1}^+(u, p; v, z) - \mu \sum_{i=1}^n \sum_{c=1}^C (q_k(i, c) - y_k(i, c)) [\delta q(i, c) / \delta w^+(u, p; v, z)]_k \quad (2)$$

$$w_k^-(u, p; v) = w_{k-1}^-(u, p; v) - \mu \sum_{i=1}^n \sum_{c=1}^C (q_k(i, c) - y_k(i, c)) [\delta q(i, c) / \delta w^-(u, p; v)]_k$$

where, $\mu > 0$ is the learning rate (some constant).

$q_k(i)$ is calculated using X_k , $w_k^+(u, p; v, z) = w_{k-1}^+(u, p; v, z)$ and

$$w_k^-(u, p; v) = w_{k-1}^-(u, p; v) \text{ in (1)}$$

$[\delta q(i, c) / \delta w^+(u, p; v, z)]_k$ and $[\delta q(i, c) / \delta w^-(u, p; v)]_k$ are evaluated using the values

$$q(i, c) = q_k(i, c), w_k^+(u, p; v, z) = w_{k-1}^+(u, p; v, z) \text{ and}$$

$$w_k^-(u, p; v) = w_{k-1}^-(u, p; v) \text{ in (2)}$$

The complete learning algorithm for the network is:

- Initiate the matrices W_0^+ and W_0^- in some appropriate manner. Choose a value of μ in (2).
- For each successive value of m :
 - Set the input-output pair (X_k, Y_k)
 - Repeat
 - Solve the equation (1) with these values
 - Using (2) and the previous results update the matrices W_k^+ and W_k^-

Until the change in the new values of the weights is smaller than some predetermined valued.

For more details about this learning algorithm, see [6].

3.2 Retrieval Procedure

Once the learning phase is completed, the network must perform as well as possible the completion of noisy versions of the training vectors. In this case, we propose a progressive retrieval process with adaptive threshold value. Let $X' = \{X'(1, 1), \dots, X'(n, C)\}$ be any input vector with values equal to 0, 0.5 or 1, for each $X'(i, c)$, $i=1, \dots, n$ and $c=1, \dots, C$. In order to determine the corresponding output vector $Y = \{Y(1, 1), \dots, Y(n, C)\}$, we first compute the vector of probabilities $Q = (q(1, 1), \dots, q(n, C))$. We consider that $q(i, c)$ values such that $1-T < q(i, c) < T/2$ or $1-T/2 < q(i, c) < T$, with for instance $T=0.8$, belong to the uncertainty interval Z . When the network stabilizes to an attractor state, the number NB_Z of neurons whose $q(i, c) \in Z$ is equal to 0. Hence, we first treat the neurons whose state is considered certain to obtain the output vector $Y^{(1)} = (Y^{(1)}(1, 1), \dots, Y^{(1)}(n, C))$, with:

$$Y^{(1)}(i, c) = Fz(q(i, c)) = \begin{cases} 1 & \text{if } q(i, c) > T \\ 0 & \text{if } q(i, c) < 1-T \\ 0.5 & \text{if } T/2 \leq q(i, c) \leq 1-T/2 \\ x'_i & \text{otherwise} \end{cases}$$

where Fz is the thresholding function by intervals. If $NB_Z = 0$, this phase is terminated and the output vector is $Y = Y^{(1)}$. Otherwise, Y is obtained after applying the thresholding function $f\beta$ as follows:

$$Y(i, c) = f\beta(q(i, c)) = \begin{cases} 1 & \text{if } q(i, c) > \beta \\ 0.5 & \text{if } \beta/2 < q(i, c) < \beta \\ 0 & \text{otherwise} \end{cases}$$

where β is the selected threshold. Each value $q(i, c) \in Z$ is considered as potential thresholds. That is, for each $q(i, c) \in Z$:

$$\beta = \begin{cases} q(i, c) & \text{if } q(i, c) > 0.666 \\ 1 - q(i, c) & \text{otherwise} \end{cases}$$

Eventually, Z can be reduced by decreasing T (for $T > 0.666$). For each potential value of β , we present to the network the vector $X^{(1)}(\beta) = f\beta(Q)$. Then, we compute the new vector of probabilities $Q^{(1)}(\beta)$ and the output vector $Y^{(2)}(\beta) = Fz(Q^{(1)}(\beta))$. We keep the cases where $NB_Z=0$ and $X^{(1)}(\beta) = Y^{(2)}(\beta)$. If these two conditions are never satisfied, the initial X' is considered too much different of any training vector. If several thresholds are candidate, we choose the one which provides the minimal error (difference between $q(i, c)$ and $Y(i, c)$, for $i=1, n$ and $c=1, \dots, C$):

$$E(\beta) = 1/2 \sum_{i=1}^n [q(i, c)^{(1)}(\beta) - Y(i, c)^{(1)}(\alpha)]^2$$

4. Experimental Results

4.1. Problem Definition

In this section, we present several examples to compare the quality of our recognition algorithm for different pattern types. We will input various geometric figures to a Multiple

Classes Random Neural Network and train the network to recognize these as separate categories. To evaluate our approach, we use three figure groups: for the first set of figures (group A), we use the set of figures shown in figure 1, where blackened boxes represent blue colors, gray boxes represent green colors and white boxes represent red colors. The next group (Group B) is composed for the black and white figures shown in figure 2, to compare our approach with the results obtained with the recognition algorithm based on RNN proposed in [3], and the evolutionary learning approach proposed in [5]. The last group is composed by the set of patterns used in [17] (see Figure 3). For the last case, extended experiments are presented to evaluate the performance of our method according to the relationship between the problem features (number of patterns, pixels and colors), recognition rates and processing time. Each pixel is represented by a neuron and we suppose three classes to represent the primary colors (red, green, and blue) according to the RGB model.

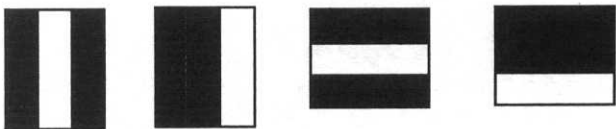


Figure 1. Geometric Figures with three colors (Group A)

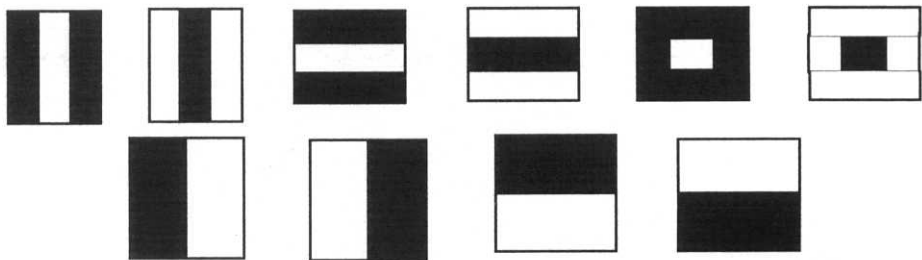


Figure 2. Geometric Figures with two colors (Group B)

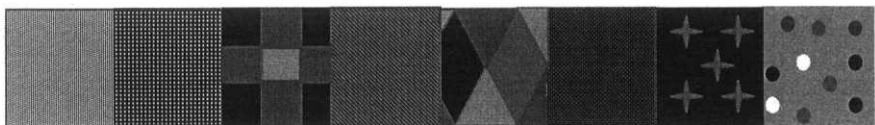


Figure 3. Pattern set used in the last experiment (Group C)

For the first and second case, each figure is represented by a 6*6 grid of pixels. For example, to represent the seventh geometric figure of the figure 2, we must use the pattern shown in figure 4. According to the RGB model, the blackened boxes are represented as (1,1,1), while white boxes are represented as (0, 0, 0). In this way, we can represent geometric figures with different combinations of colors (for example, for figure 2 if we suppose blackened boxes correspond to red colors, and white boxes to blue colors, neurons for blackened boxes are equal to (1, 0, 0) and for white boxes to (0, 0, 1)). Thus, for these cases we use a single-layer multiple classes RNN composed by 36 neurons ($n=36$) and 3 classes ($C=3$).

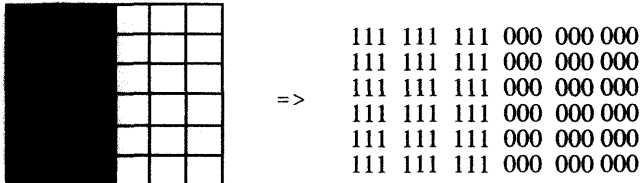


Figure 4. Representation of a geometric figure with a 6*6 pattern

4.2. Results Analysis

In order to test associative memories, we have evaluated the recognition rates of distorted versions of the training patterns (tables 1 and 2). We generated 20 noisy images used as inputs, for each training image and for a given distortion rate. The result of the learning stage is used as the initial neural network of this second phase (retrieval stage). We have corrupted them by reasonable noise rates equal to 0%, 10%, 20% and 30% distortion by modifying bit values at random. A pattern is recognized if the residual error rate is less than 3%. The results we have obtained are presented on table 1 and 2. These values represent the average of 8 processes for each set S_i of images. The performance results obtained are lower when the noise rate is important (memories are then more discriminating). The results for the first group are presented on table 1. Our algorithm provides a good recognition rate. Particularly, the recognition rate of the sets S_4 and S_6 remain good for our approach. Concerning S_{10} and 30% of noise rate, recognition rate decreases.

Noisy Rate	0%			10%			20%			30%		
Number of Figures	4	6	10	4	6	10	4	6	10	4	6	10
Group A	99%	99%	97%	94%	93%	89%	83%	82%	81%	73%	71%	67%

Table 1. Recognition rate of noisy versions of Group A

In table 2 are shown the recognition rate for the last group of images (Group B), using the classical gradient recognition algorithm (*Cl*), the hybrid Genetic/Random Neural Network learning algorithm (*Evol*) and our Multiple Classes learning algorithm (*Mult*). In general, *Evol* appears to give the best results. The recognition rate remains good for our algorithm (*Mult*). This algorithm provides a better recognition rate than *Cl*. Concerning *Cl*, the recognition rate is bad.

Noisy Rate	0%			10%			20%			30%		
Number of Figures	4	6	10	4	6	10	4	6	10	4	6	10
Cl	99%	95%	96%	93%	91%	85%	83%	80%	77%	68%	66%	62%
Evol	99%	99%	99%	96%	95%	92%	87%	85%	81%	75%	74%	72%
Mult	99%	99%	99%	95%	94%	90%	85%	84%	81%	73%	72%	70%

Table 2. Recognition rate of noisy versions of Group B

In figure 5 are shown the system errors during the learning phase for the Group B, using the classical gradient decent learning algorithm (*Cl*), the hybrid Genetic/Random Neural Network learning algorithm (*Evol*) and our Multiple Classes learning algorithm (*Mult*). *Evol* gives the best results, but with a substantially large execution time. That is because *Evol* is very slow to converge. The learning error remains good for our learning algorithm (*Mult*). This algorithm provides a better error convergence of the learning phase than *Cl*. Concerning *Cl*, error costs are important, that is the reason for its bad recognition rate.

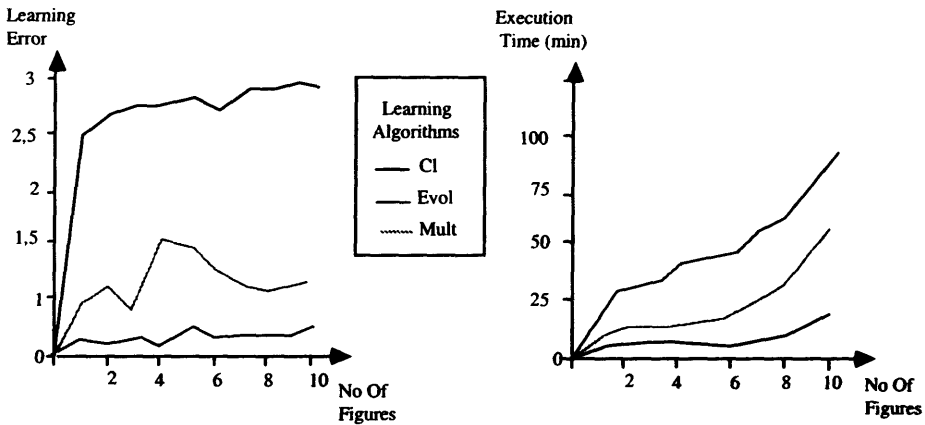


Figure 5. Learning error and execution time of the learning algorithms for Group B

In table 3 are shown the relationship between the problem features (number of pixels and colors), recognition rates and processing time for the set of figures of the group C. If we increase the number of pixel to describe a pattern, we improve the quality of the retrieval phase, but the execution time increases exponentially. The number of colors is not important because our system doesn't depend on it. If the RGB model can represent the specific color of a given pattern, we approach can recognized it (see the similarity of the recognition rates for the cases of the figures 1-3 and 6-8). When the patterns are different (patterns 6, 7 and 8 of the figure 3) the system has a better retrieval rate. Our approach can recognize several numbers of patterns, but with a large retrieval time if we like to obtain goods retrieval rates.

Our system has a typical associative memory approach problem: low storage capability. If we compare the quality of our results with the method proposed on [8, 17], their approach has a better storage capability (they tested their approach for 30 patterns), but our recognition rate quality is better ($\geq 90\%$ for 20% of noisy rate).

Noisy Rate	0%				10%				20%			
Figures	1-8	1-3	6-8	1-5	1-8	1-3	6-8	1-5	1-8	1-3	6-8	1-5
Number of pixels	144	144	144	144	144	144	144	144	144	144	144	144
Recognition Rates	95%	98%	98%	94%	88%	92%	95%	90%	78	83%	84%	80%
Retrieval Time (sec)	240	31	33	120	300	34	43	60	234	33	38	55
Number of pixels	576	576	576	576	576	576	576	576	576	576	576	576
Recognition Rates	95%	99%	99%	95%	89%	93%	96%	91%	79%	85%	86%	82%
Retrieval Time	720	120	123	212	723	114	132	221	756	134	144	321
Number of pixels	2304	2304	2304	2304	2304	2304	2304	2304	2304	2304	2304	2304
Recognition Rates	99%	99%	99%	99%	95%	97%	97%	96%	92%	95%	95%	92%
Retrieval Time	9188	1302	1287	2341	8923	1100	1098	2178	9012	1231	1101	2111

Table 3. Performance Evaluation of our method with the noisy versions of Group C

5. Conclusions

In this paper, we have propose a recognition algorithm based on the Multiple Classes Random Neural Model. We have shown that this model can efficiently work as associative memory. We can recognize arbitrary color images with this algorithm, but the processing time will increase rapidly according to the number of pixels used. The number of neurons is dictated by the image resolution, and it has a direct influence on the quality of the

performance of our approach. During the learning phase, we have met classical problems like the existence of local minimal and large learning times. At level of retrieval algorithm, we have obtained good performance but with a large execution time. However, most of the computations are intrinsically parallel and can be implemented on SIMD or MIMD architectures. At this moment we work in a data parallel version of our approach.

Acknowledgment

This work was partially supported by FONACIT grant AP-97003817, CDCHT-ULA grant I I-621-98-02-A and CeCaCULA (High Performance Computing Center of Venezuela).

References

- [1] J. Aguilar, Evolutionary Learning on Recurrent Random Neural Network, World Congress on Neural Networks, 1995, pp. 232-236.
- [2] J. Aguilar, An Energy Function for the Random Neural Networks, *Neural Processing Letters*, **4** (1996) 17-27.
- [3] J. Aguilar, A Recognition Algorithm using the Random Neural Network, 3rd. International Congress on Computer Science Research, 1996, pp. 15-22.
- [4] J. Aguilar, Definition of an Energy Function for the Random Neural to solve Optimization Problems, *Neural Networks*, **11** (1998) 731-738.
- [5] J. Aguilar, A. Colmenares, Resolution of Pattern Recognition Problems using a Hybrid Genetic/Random Neural Network Learning Algorithm, *Pattern Analysis and Applications*, **1** (1998) 52-61.
- [6] J. Aguilar, Learning Algorithms for the Multiple Classes Random Neural Network, *Lecture Notes in Artificial Intelligence*, **1821** (2000) 561-566.
- [7] V. Atalay, E. Gelenbe, N. Yalabik, The random neural network model for texture generation, *Intl. Journal of Pattern Recognition and Artificial Intelligence*, **6** (1992) 131-141.
- [8] C. Colombo, A. Del Bimbo, Color-induced image representation and retrieval, *Pattern Recognition*, **32** (1999) 1685-1695.
- [9] A. Del Bimbo, P. Pala, Visual image retrieval by elastic matching of user sketches, *IEEE Trans. Pattern Anal. Machine Intelligence*, **19** (1997) 223-234.
- [10] M. Fourneau, E. Gelenbe, R. Suros, G-networks with Multiple classes of negative and positive customers, *Theoretical Computer Science*, **155** (1996) 141-156.
- [11] E. Gelenbe, Random neural networks with positive and negative signals and product form solution, *Neural Computation*, **1** (1989) 502-511.
- [12] E. Gelenbe, Stability of the random neural networks, *Neural Computation*, **2** (1990) 239-247.
- [13] E. Gelenbe, Theory of the random neural network model, In E. Gelenbe (ed.), *Neural Networks: Advances and Applications*, North-Holland, Pays-Bas, 1991.
- [14] E. Gelenbe, Learning in the recurrent random neural network, *Neural Computation*, **5** (1993) 376-389.
- [15] A. Jain, A. Vailaya, Image Retrieval using Color and Shape, *Pattern Recognition*, **29** (1996) 1233-1244.
- [16] T. Minka, R. Picard, Interactive Learning with a Society of Models, *Pattern Recognition*, **30** (1997) 1370-1381.
- [17] A. Mojsilovic, J. Kovacevic, D. Kall, R. Safranek, K. Ganapathy, The Vocabulary and Grammar of Color Patterns, *IEEE Trans. on Image Processing*, **9** (2000) 417-431.

Jigsawing : A Method to Create Virtual Examples in OCR data

S.V.N. Vishwanthan, M. Narasimha Murty*

{ vishy, mnm }@csa.iisc.ernet.in

Dept. Of Comp. Sci. and Automation

Indian Institute of Science

Bangalore 560 012, India

Abstract. In this theoretical note we propose the use of a suffix tree on square matrices for compact representation of a set of training patterns. We show how a test pattern can be generated by *jigsawing* various regions from different training patterns. This in turn leads us naturally to a compact data dependent representation of a test pattern which we call the *description tree*. We envisage the use of the description tree in a variety of applications including nearest neighbor classifiers, data dependent distance norms, kernel methods and syntactic pattern recognition. We provide statistical learning theory based arguments to show that our method generates valid virtual examples and hence will lead to better classification accuracy.

1 Introduction

One of the fundamental problems in supervised learning is the lack of an adequate number of training data [7]. In many real life applications we have some domain knowledge about the training set which can be used to generate virtual training samples. Some successful attempts in this direction especially for OCR data sets include methods like bootstrapping and Partitioned Pattern Classification (PPC) trees [12].

Bootstrapping generates new data points by either taking the centroid or the medoid of a set of *closely related* points. A trie is a multi-way tree structure used for storing strings over an alphabet. The PPC tree partitions the dataset vertically into blocks and constructs a separate trie for each block of each class of training samples [12]. The main difference between bootstrapping and a PPC tree is that the former produces new training data explicitly while the PPC tree can generate new samples implicitly.

The advantage of the PPC tree is that it uses linear space to encode an exponential number of virtual samples. But the main disadvantage is that the partitioning into blocks is highly dataset dependent. Besides it does not generalize well to take into consideration connected regions of a pattern. This is especially important for OCR data sets where the structure of the data plays a major role in classification algorithms. As of now the PPC tree has been used only with a KNN classifier.

*Corresponding author

The suffix tree is a compacted trie that stores all suffixes of a given text string [13, 6]. It has been widely used for compact representation of input text and in a wide variety of search applications [4]. The notion of a suffix tree on a string can also be generalized to a suffix tree on a matrix [3]. Given an $n \times n$ square matrix A , whose entries are drawn from an ordered alphabet Σ , we build an compacted trie. For each square sub-matrix of A , there is a path in the trie which corresponds to that sub-matrix. Linear time $O(n^2)$ and linear space $O(n^2)$ algorithms exist for constructing such suffix trees [5, 1]. The suffix tree of a set of matrices \mathcal{A} can be built by merging the suffix trees of individual elements of \mathcal{A} in linear time. We denote such a suffix tree by $S(\mathcal{A})$. Searching for all occurrences of a matrix B of size $m \times m$ can be done in $O(m^2)$ time [3]. We use the notation $B \sqsubseteq \mathcal{A}$ to denote that B occurs as a sub-matrix of some element of \mathcal{A} .

An immediately apparent application of the suffix tree on matrices is to search for the occurrence of a test pattern in the training set [3]. In real world applications the probability of finding an *exact* match between training patterns and a test pattern is negligible. This limits the applicability of the 2-d suffix trees. It is clear that some scheme which allows approximate matches is required to tackle this problem.

We cast the problem of finding approximate matches as that of extracting a measure of similarity. We achieve this by recursively sub-dividing the test pattern in order to find a match. As a result different regions of the test pattern may match regions from different training patterns. Hence, a test pattern can be *described* as a *jigsaw* of regions from various training patterns. The weightage assigned to a matching region depends on its size, number of times it occurs in the training set and its location coordinates in the training sample. For example if a large region of the test pattern matches with corresponding regions (occurring at the same location coordinates) of the training patterns of a given class then we assign a high weight value to such a match.

Our paper is organized as follows. In section 2 we introduce the reader to a few statistical learning theory tools and describe the need to produce virtual examples. We describe our algorithm in detail in section 3. We discuss the generation of the description tree of a test pattern and also discuss various applications which can benefit from such a representation. We also discuss the algorithm in the light of statistical learning theory tools. We conclude in section 4 with pointers to future research and a few open problems.

2 Statistical Learning Theory

In this section we provide a brief introduction to statistical learning theory and motivate the necessity for generating virtual examples.

2.1 The Learning Problem

We begin by introducing some notation. Assume that we are given a training set of m labelled patterns

$$\mathcal{X} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$$

where $\mathbf{x}_i \in \mathcal{R}^{n \times n}$ are the patterns and $y_i \in \{+1, -1\}$ are the corresponding labels¹. Furthermore, we assume that $n = 2^l$ for some $l \in \mathcal{N}$ ². We denote by \mathcal{X}_+ the training samples belonging to class +1 and by \mathcal{X}_- the samples belonging to class -1. We also assume that the training samples are all drawn independent and identically distributed from an unknown probability distribution $P(\mathbf{x}, y)$.

A learning algorithm is associated with a class of hypothesis which it can implement. We denote the class of hypothesis by \mathcal{H} . The members of \mathcal{H} are assumed to be parameterized by a set of adjustable parameters α and the learning function generated is denoted by $f(\mathbf{x}, \alpha) : \mathcal{R}^n \rightarrow \{+1, -1\}$. Our aim is to pick a function from \mathcal{H} which closely models $P(\mathbf{x}, y)$. The hope is that given a rich enough class \mathcal{H} and a *adequate* number of training samples the learning algorithm will be able to *generalize* and predict well on future unseen samples.

2.2 Vapnik-Chervonenkis Bounds

The empirical risk for a learning machine is just the measured mean error rate on the training set. The 0 – 1 loss function incurs a unit loss for every misclassified sample and does not penalize correctly classified samples. Using such a loss function the empirical risk can be written as

$$R_{\text{emp}}(\alpha) = \frac{1}{2m} \sum_{i=1}^m |f(\mathbf{x}_i, \alpha) - y_i|$$

while the actual risk can be written as

$$R_{\text{actual}}(\alpha) = \int \frac{1}{2} |f(\mathbf{x}, \alpha) - y| dP(\mathbf{x}, y)$$

For $\eta \in (0, 1)$ the following bound holds with probability $1 - \eta$ [9].

$$R_{\text{actual}}(\alpha) \leq R_{\text{emp}}(\alpha) + \phi \left(\frac{h}{m}, \frac{\log(\eta)}{m} \right) \quad (1)$$

where ϕ is called the *confidence* term. Here, h is defined to be a non negative integer called the Vapnik Chervonenkis (VC) dimension. Loosely speaking the VC-dimension of a machine measures the richness of the function class \mathcal{H} [8]. It is clear from Equation (1) that minimizing the bound involves minimizing the empirical risk as well as the VC-dimension of the learning machine.

2.3 Generating Virtual Examples

It is well known that the empirical risk of a classifier can be reduced by training it on a large number of samples [2]. In many real life situations the data set size may be limited or it may be expensive to obtain new data points for training. But, we usually have some prior domain knowledge about the data which can be used to generate synthetic patterns. Let $T : D^t \rightarrow D$

¹ An extension to multiple classes is straightforward. We consider the binary classification problem only for notational convenience

² If this condition is not satisfied we can always pad each \mathbf{x}_i with an adequate number of rows and columns of zeros.

be a transformation which maps a set of t training samples drawn from domain D into a new training pattern in the same domain. New meaningful training patterns can be generated by applying the transformation T to the elements of the training set \mathcal{X} [7]. A simple example of such a transformation is the technique of bootstrapping wherein new data points are generated by either taking the centroid or mediod of a set of *closely related* points.

Our algorithm tries to decrease the empirical risk by creating new samples using a new technique called jigsawing. We implicitly generate new data points implicitly by combining various regions of different training samples. In the next section we describe our algorithm in detail and discuss its effectiveness using tools from statistical learning theory.

3 Our Algorithm

Given a pattern $\mathbf{x} \in \mathbb{R}^{n \times n}$, the numbers $a, b, l \in \mathbb{R}$ such that $1 \leq a, b \leq n$ and $1 \leq a+l, b+l \leq n$ define a square sub-matrix $\mathbf{x}[a : a+l, b : b+l]$. Let $\text{submat}_{\mathbf{x}}(., ., .)$ be a function such that $\text{submat}_{\mathbf{x}}(a, b, l) := \mathbf{x}[a : a+l, b : b+l]$. We define a function $\text{split}_{\mathbf{x}}(a, b, 2l)$ which subdivides $\text{submat}_{\mathbf{x}}(a, b, 2l)$ into four square sub-matrices given by

$$\begin{aligned} R_1 &= \text{submat}_{\mathbf{x}}(a, b, l) \\ R_2 &= \text{submat}_{\mathbf{x}}(a, b+l+1, l) \\ R_3 &= \text{submat}_{\mathbf{x}}(a+l+1, b, l) \\ R_4 &= \text{submat}_{\mathbf{x}}(a+l+1, b+l+1, l). \end{aligned}$$

Let, T be a threshold parameter such that $T \leq l$, $\lambda \in (0, 1)$ be a weighting factor and $t \in \mathbb{R}^{n \times n}$ be a test pattern. A description tree of t wrt samples \mathcal{X}_c (belonging to class c) is denoted by $D(t, \mathcal{X}_c)$. It is a weighted tree with maximum depth T such that the weight of an internal node is given by the sum of the weights of all the leaves hanging off the subtree rooted at that node. The weight on the root node gives a measure of similarity between t and \mathcal{X}_c . If $t \in \mathcal{X}_c$ then $D(t, \mathcal{X}_c)$ consist of the a single root node which is assigned a maximum possible weight of 1.

Algorithm 1 SplitNode

```

input  $R, d, N$ 
if  $d \geq T$  then
     $\text{weight}(N) = 0$ 
    return
end if
Add  $\{N_1, N_2, N_3, N_4\}$  as children of  $N$ 
for  $R_i \in \text{split}(R)$  do
    if  $R_i \subseteq \mathcal{X}_c$  then
         $\text{weight}(N_i) = (\lambda/4)^d$ 
    else
        SplitNode( $R_i, d+1, N_i$ )
    end if
end for

```

Our algorithm begins by constructing the two dimensional suffix trees $S(\mathcal{X}_+)$ and $S(\mathcal{X}_-)$ in linear time and linear space [1]. It then recursively calls the procedure shown in Algorithm 1. Given a square sub-matrix of t denoted by R , a node N in $D(t, \mathcal{X}_c)$ at depth d the procedure splits R into four equal sized square sub-matrices. Four nodes are added to $D(t, \mathcal{X}_c)$ at depth $d + 1$ in order to represent these new regions. Each of the sub-matrices is checked for an exact match in \mathcal{X}_c . The recursive procedure is invoked either until an exact match is found or the threshold depth T is reached.

Our algorithm first looks for an exact match for t in \mathcal{X} . If such a match is not found the recursive procedure is invoked to split t into sub-regions. At each level the sum of the sizes of regions considered for matching is bound by $n \times n$. Hence the work done per level of the description tree is never more than $O(n^2)$. Since the number of levels is bounded by $T \leq \log_2(n)$ the whole description tree construction takes at most $O(n^2 \log_2(n))$ time. The cost of constructing the suffix tree $S(\mathcal{X}_c)$ is a one time $O(|\mathcal{X}_c|)$ cost while the construction of the description tree $D(t, \mathcal{X}_c)$ is independent of the size of \mathcal{X}_c and depends only on the size of the test pattern. This is significantly cheaper than the $O(n^2 |\mathcal{X}_c|)$ cost incurred by the nearest neighbor classifier [2]. The outline of our algorithm is shown in Algorithm 2.

Algorithm 2 Description Tree Construction

```

input   $t, S(\mathcal{X}_c), T, \lambda$ 
output  $D(t, \mathcal{X}_c)$ 
  Let  $D(t, \mathcal{X}_c) \leftarrow \text{root}$ 
  if  $t \subseteq \mathcal{X}_c$  then
     $\text{weight}(\text{root}) = 1$ 
  else
     $\text{SplitNode}(t, 1, \text{root})$ 
  end if

```

3.1 Statistical Learning Theory and Jigsawing

We know that, in general, OCR patterns preserve *structural* properties. Given two training samples x_i and x_j drawn from the same class c let R_{x_i} be a region in x_i and R_{x_j} be the corresponding region in x_j . New (possibly valid) training patterns can be obtained by swapping R_{x_i} and R_{x_j} . Because of the structure preserving nature of OCR data we expect that the new sample would be valid (or at least close to valid) patterns of class c . The jigsawing algorithm described above swaps square sub-matrices to produce new samples.

In real life situations all such samples produced by swapping various regions may not be valid. For example as we jigsaw smaller and smaller square sub-matrices (i.e. we allow larger and larger values of the threshold T) the number of virtual examples we generate grows exponentially. While many of these points are meaningful points the number of noisy or meaningless points that are produced also increases. This in turn may result in more misclassification and hence increase the empirical risk.

We must also keep in mind that as the effective training set size increases a more complex hypothesis class \mathcal{H} may be needed to effectively *explain* the training data [9]. This in turn may mean that we need a classifier with higher VC dimension. The VC dimension of classifiers like that the KNN classifier have an implicit dependence on the training set size and

complexity. Hence, as the number of virtual examples increases the confidence term of the classifier also increases and leads to looser bounds on the actual risk.

As a result of these two effects the classification accuracy tends to increase when jigsawing is done with smaller values of T . But as the value of T increases the classification accuracy may come down.

3.2 Applications

In this section we try to give a flavor for the various applications of the description tree algorithm we described above. It is straightforward to design a classifier given the description trees $D(t, \mathcal{X}_+)$ and $D(t, \mathcal{X}_-)$. The weight of all the internal nodes can be found by doing a simple depth first search on $D(x, \mathcal{X}_c)$. The weights on the respective root nodes can be used to assign t to the class with maximum similarity.

The description tree $D(t, \mathcal{X}_c)$ can be used in Support Vector Machines [8] and kernel methods using ideas outlined in [11]. This can lead to other exciting ideas of data dependent kernels and projections into higher dimensional spaces. The description tree is a true data dependent measure of similarity and hence can be used for clustering applications. It can also be used to evaluate the discriminating capability of a training set by looking at the dissimilarity measure between the trees $D(t, \mathcal{X}_+)$ and $D(t, \mathcal{X}_-)$ for different test patterns.

Using a leave one out procedure the description tree of a training pattern can be constructed using the rest of the training samples from the same class. The depth and branching levels of this tree can be used for data set reduction. A pattern which generates a highly branched and deep description tree is in some sense very dissimilar to other patterns already present in the training set and hence must be retained. On the other hand a training pattern which produces a shallow description tree with a large weight on the root node is very similar to already existing patterns in the training set and hence can be pruned. Such ideas can be effectively combined with other data set reduction techniques like those proposed in [10].

4 Conclusion

We have proposed a novel method to compactly represent the 2-d structural properties of OCR data using a suffix tree on matrices. This representation leads to a true data dependent similarity measure for a test pattern which can be computed much faster than the distance of a new data point from each point in the training set. Many applications of the above ideas especially in the field of pattern recognition, machine learning and OCR data recognition are current topics of research and will be reported in subsequent publications.

5 Acknowledgements

S.V.N. Vishwanathan is supported by an Infosys Fellowship. The authors would like to thank P. Viswanath for useful discussions.

References

- [1] R. Cole and R. Hariharan. Faster suffix tree construction with missing suffix links. In *Proceedings of the Thirty Second Annual Symposium on the Theory of Computing*, Portland, OR, USA, May 2000. ACM.

- [2] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York, 2001. Second edition.
- [3] R. Giancarlo. A generalization of the suffix tree to matrices with applications. *SIAM Journal on Computing*, 24(3):520–562, 1995.
- [4] R. Grossi and G. Italiano. Suffix trees and their applications in string algorithms. In *Proceedings of 1st South American Workshop on String Processing (WSP 1993)*, pages 57 – 76, September 1993.
- [5] D.K. Kim and K. Park. Linear-time construction of two-dimensional suffix trees. In *Proceedings of the 26th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 1644 of *LNCS*, pages 463–472, Prague, Czech, July 1999. Springer.
- [6] E. M. McCreight. A space-economical suffix tree construction algorithm. *Journal of the ACM*, 23(2):262 – 272, April 1976.
- [7] P. Niyogi, F. Girosi, and T. Poggio. Incorporating prior knowledge in machine learning by creating virtual examples. *Proceedings of IEEE*, 86(11):2196 – 2209, November 1998.
- [8] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- [9] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.
- [10] S.V.N. Vishwanathan and M. Narasimha Murty. Use of MPSVM for data set reduction. In A. Abraham and M. Koeppen, editors, *Hybrid Information Systems*, Heidelberg, 2002. Physica Verlag.
- [11] S.V.N. Vishwanathan and A.J. Smola. Fast kernels on strings and trees. In *Proceedings of NIPS*, 2002. submitted.
- [12] P. Viswanath and M. Narasimha Murty. An efficient incremental mining algorithm for compact realization of prototypes. Technical Report IISC-CSA-2002-2, CSA Department, Indian Institute of Science, Bangalore, India, January 2002.
- [13] P. Weiner. Linear pattern matching algorithms. In *Proceedings of the IEEE 14th Annual Symposium on Switching and Automata Theory*, pages 1 – 11, The University of Iowa, 1973. IEEE.

Model-Based Restoration of Short-Exposure Solar Images

Michal Haindl and Stanislava Šimberová*
Institute of Information Theory and Automation,
Academy of Sciences,
Prague, CZ182 08,
*Astronomical Institute**, Academy of Sciences,
Ondřejov, CZ251 65, Czech Republic

Abstract. This paper presents a derivation of a fast recursive filter for image restoration if degradation obeys a linear degradation model with the unknown possibly non-homogeneous point-spread function. It is assumed that for every ideal undegraded image several degraded observed images are available. Pixels in the vicinity of steep discontinuities are left unrestored to minimize restoration blurring effect. The degraded image is assumed to follow a causal simultaneous multidimensional regressive model and the point-spread function is estimated using the local least-square estimate.

1 Introduction

Real imaging systems, the recording medium, the atmosphere are imperfect and thus a recorded image represents a degraded version of the original scene. Similarly an image is usually further corrupted during its processing, transmission or storage. Possible examples are lens defocusing or aberration, noisy transmission channels, motion between camera and scene, etc.

There are two main types of the image degradation obtained by the ground-based telescopes. First, the transfer of information through the optical system is not perfect. The point spread function (PSF) of the telescope embodies all the important behaviour of the optical image formation system. For a specific telescope with given optical parameters and the spectral wavelength the PSF can be estimated. Second, much more difficult part represents degradation by the turbulence of the Earth's atmosphere. The major degradation of a ground-based telescope is caused by random fluctuations of the optical way between the object space (the Sun) and the image formation device (telescope). The image quality in the image space decreases by variations of the index of refraction, dominated by the thermal turbulence. This effect is called "seeing". Influence of the turbulence in image formation were analyzed since the 1950s, see e.g. [6],[22],[23]. Many efficient methods for image restoration have been developed and many sophisticated algorithms in the last 10 years have been applied. In ground-based solar observations it is often possible to obtain several images of the object under investigation that differ just by the component of PSF originating from seeing. For the reconstruction we suppose a short time sequence (max. to 30s) of images. The exposure time of the each image is $< 20\text{ms}$, so we are allowed to involve them into the "short-exposed" ones and suppose a still scene in the image. Our approach to the image restoration is different to

direct and indirect techniques like various types of filtering, power spectral equalization, constrained least-squares restoration, maximum entropy restoration, etc. The image restoration task is to recover an unobservable image given the observed corrupted images with respect to some statistical criterion.

The simplest restoration method is to smooth the data with an isotropic linear or non-linear shift-invariant low-pass filter. Usual filtering techniques (e.g. median filter, Gaussian low pass filter, band pass filters, etc.) tend to blur the location of boundaries. Several methods [19] try to avoid this problem by using a large number of low-pass filters and combining their outputs. Similarly anisotropic diffusion [20],[5] addresses this problem but it is computationally extremely demanding. Image intensity in this method is allowed to diffuse over time, with the amount of diffusion at a point being inversely proportional to the magnitude of local intensity gradient. A nonlinear filtering method developed by Nitzberg and Shiota [18] uses an offset term to displace kernel centers away from presumed edges and thus to preserve them, however it is not easy to propose all filter parameters to perform satisfactory on variety of different images and the algorithm is very slow. In the exceptional case, when the degradation point-spread function is known, the Wiener filter [1] or deconvolution methods [14] can be used. Model-based methods use most often Markov random field type of models either in the form of wide sense Markov (regressive models) or strong Markov models. The noncausal regressive model used in [3],[4] has the main problem in time consuming iterative solution based on the conjugate gradient method. Similarly Markov random field based restoration methods [8], [7], [15] require time consuming application of Markov chain Monte Carlo methods. Besides this both approaches have solve the problem when to stop these iterative processes. A similar combination of causal and non-causal regressive models as in this paper was used in [16]. However they assume the homogeneous point-spread function and they identify all parameters simultaneously using extremely time consuming iterations of the EM (expectation maximization) algorithm which is not guaranteed to reach the global optimum. This work generalizes our monospectral restoration method [9],[13] for the multiversion images. It is seldom possible to obtain a degradation model analytically from the physics of the problem. More often a limited prior knowledge supports only some elementary assumptions about this process. Usual assumption, accepted also in this work, is that the corruption process can be modeled using a linear degradation model.

The section 2 introduces the image degradation model and the core part of the restoration algorithm. The section 3 contains the model selection criterion, while the following section 4 represents the PSF estimation. The results and conclusions are in sections 5 and 6 given.

2 Image Model

Suppose Y represents a true but unobservable monospectral image defined on the finite rectangular $N \times M$ underlying lattice I . Suppose further that we have a set of d observable images \mathcal{X} where each $X_{\bullet,i} \in \mathcal{X}$ is the i -th version of Y distorted by the unknown PSF and noise independent of the signal. The notation \bullet has the meaning of all possible values of the corresponding multiindex (e.g. the multiindex $r = \{r_1, r_2\}$ which has the row and columns indices, respectively). We assume knowledge of all pixels elements from the reconstructed scene. For the treatment of the more difficult problem when some data are missing see [11], [12]. The image degradation is supposed to be approximated by the linear discrete spatial domain degradation model

$$X_{r,\bullet} = \sum_{s \in I_r} H_s Y_{r-s} + \epsilon_{r,\bullet} \quad (1)$$

where H is a discrete representation of the unknown point-spread function, $X_{r,\bullet}$ is the $d \times 1$ vector of the r -th pixel in different distortions and Y_{r-s} are ideal (unobservable) image pixels. The point-spread function is assumed to be either homogeneous or it can be non-homogeneous but in this case we assume its slow changes relative to the size of an image. I_r is some contextual support set, and a noise vector ϵ is uncorrelated with the true image, i.e.,

$$E\{Y \epsilon_{\bullet,i}\} = 0 \quad (2)$$

The point-spread function is unknown but such that we can assume the unobservable image Y to be reasonably well approximated by the expectation of the corrupted image

$$\hat{Y} = E\{X_{\bullet,i}\} \quad (3)$$

in regions with gradual pixel value changes and i -th degraded image $X_{\bullet,i} \in \mathcal{X}$ is the least degraded image from the set \mathcal{X} . The index i of the least degraded image is excluded from equations (4)-(9), (12)-(24) to simplify corresponding notation. The above method (3) changes all pixels in the restored image and thus blurs discontinuities present in the scene although to much less extent than the classical restoration methods due to adaptive restoration model (10). This excessive blurring can be avoided if pixels with steep step discontinuities are left unrestored, i.e.,

$$\hat{Y}_r = \begin{cases} E\{X_r\} & \text{if (5) holds} \\ X_r & \text{otherwise} \end{cases} \quad (4)$$

where the adaptive condition (5) is

$$|E\{X_r\} - X_r| < \frac{1}{n_s} \sum_s |E\{X_{r-s}\} - X_{r-s}| \quad (5)$$

The expectation (3) can be expressed as follows:

$$E\{X\} = \int x p(x) dx = \int \begin{pmatrix} x_1 & x_2 & \dots & x_M \\ x_{M+1} & x_{M+2} & \dots & x_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ x_{NM-M+1} & x_{NM-M+2} & \dots & x_{NM} \end{pmatrix} \prod_{r=1}^{NM} p(x_r | X^{(r-1)}) dx_1 \dots dx_{NM} \quad (6)$$

where $X^{(r-1)} = \{X_{r-1}, \dots, X_1\}$ is a set of noisy pixels in some chosen but fixed ordering. For single matrix elements in (6) it holds

$$\begin{aligned}
E\{X_j\} &= \int x_j \prod_{r=1}^{NM} p(x_r | x^{(r-1)}) dx_1 \dots dx_{NM} = \int X_j \prod_{r=1}^j p(X_r | X^{(r-1)}) dX_1 \dots dX_j \\
&= \int E\{X_j | X^{(j-1)}\} \prod_{r=1}^{j-1} p(X_r | X^{(r-1)}) dX_1 \dots dX_{j-1} \\
&= E_{X^{(j-1)}}\{E_{X_j}\{X_j | X^{(j-1)}\}\}
\end{aligned} \tag{7}$$

Let us approximate after having observed $x^{(j-1)}$ the $\hat{Y}_j = E\{X_j\}$ by the conditional expectation $E\{X_j | X^{(j-1)} = x^{(j-1)}\}$ where $x^{(j-1)}$ are known past realization for j . Thus we suppose that all other possible realization $x^{(j-1)}$ than the true past pixel values have negligible probabilities. This assumption implies conditional expectations approximately equal to unconditional ones, i.e., then the expectation (7) is

$$E\{X_j\} \approx E\{X_j | X^{(j-1)}\}, \tag{8}$$

and

$$\hat{Y} = E\{X\} \approx \begin{pmatrix} E\{X_1 | x^{(0)}\} & \dots & E\{X_M | x^{(M-1)}\} \\ E\{X_{M+1} | x^{(M)}\} & \dots & E\{X_{2M} | x^{(2M-1)}\} \\ \vdots & \ddots & \vdots \\ E\{X_{NM-M+1} | x^{(NM-M)}\} & \dots & E\{X_{NM} | x^{(NM-1)}\} \end{pmatrix} \tag{9}$$

Suppose further that a noisy image can be represented by an adaptive 2.5D causal simultaneous autoregressive model

$$X_{r,i} = \sum_{s \in I_r^c} A_s X_{r-s, \bullet} + \epsilon_r, \tag{10}$$

where ϵ_r is a white Gaussian noise vector with zero mean, and a constant but unknown covariance matrix Σ . The noise vector is uncorrelated with data from a causal neighbourhood I_r^c .

$$A_s = [a_{s,1}, \dots, a_{s,d}] \quad \forall s$$

are parameter vectors. The model adaptivity is introduced using the standard exponential forgetting factor technique in parameter learning part of the algorithm. The model can be written in the matrix form

$$X_{r,i} = \gamma Z_r + \epsilon_r, \tag{11}$$

where

$$\gamma = [A_1, \dots, A_\eta], \tag{12}$$

$$\eta = \text{card}(I_r^c) \tag{13}$$

is a $1 \times d\eta$ parameter matrix and Z_r is a corresponding vector of X_{r-s} . To evaluate conditional mean values in (9) the one-step-ahead prediction posterior density $p(X_r | X^{(r-1)})$ is

needed. If we assume the normal-gamma parameter prior for parameters in (10) (alternatively we can assume the Jeffrey's parameter prior) this posterior density has the form of Student's probability density

$$p(X_r|X^{(r-1)}) = \frac{\Gamma(\frac{\beta(r)-d\eta+3}{2})}{\Gamma(\frac{\beta(r)-d\eta+2}{2}) \pi^{\frac{1}{2}} (1 + Z_r^T V_{z(r-1)}^{-1} Z_r)^{\frac{1}{2}} \lambda_{(r-1)}^{\frac{1}{2}}} \left(1 + \frac{(X_r - \hat{\gamma}_{r-1} Z_r)^T \lambda_{(r-1)}^{-1} (X_r - \hat{\gamma}_{r-1} Z_r)}{1 + Z_r^T V_{z(r-1)}^{-1} Z_r} \right)^{-\frac{\beta(r)-d\eta+3}{2}}, \quad (14)$$

with $\beta(r) - d\eta + 2$ degrees of freedom, where the following notation is used:

$$\beta(r) = \beta(0) + r - 1 = \beta(r-1) + 1, \quad (15)$$

$$\beta(0) > 1,$$

$$\hat{\gamma}_{r-1}^T = V_{z(r-1)}^{-1} V_{zx(r-1)}, \quad (16)$$

$$V_{r-1} = \tilde{V}_{r-1} + I,$$

$$\tilde{V}_{r-1} = \begin{pmatrix} \tilde{V}_{x(r-1)} & \tilde{V}_{zx(r-1)}^T \\ \tilde{V}_{zx(r-1)} & \tilde{V}_{z(r-1)} \end{pmatrix}, \quad (17)$$

$$\tilde{V}_{x(r-1)} = \sum_{k=1}^{r-1} X_k X_k^T, \quad (18)$$

$$\tilde{V}_{zx(r-1)} = \sum_{k=1}^{r-1} Z_k X_k^T, \quad (19)$$

$$\tilde{V}_{z(r-1)} = \sum_{k=1}^{r-1} Z_k Z_k^T, \quad (20)$$

$$\lambda_{(r)} = V_{x(r)} - V_{zx(r)}^T V_{z(r)}^{-1} V_{zx(r)}. \quad (21)$$

If $\beta(r-1) > \eta$ then the conditional mean value is

$$E\{X_r|X^{(r-1)}\} = \hat{\gamma}_{r-1} Z_r \quad (22)$$

and it can be efficiently computed using the following recursion

$$\hat{\gamma}_r^T = \hat{\gamma}_{r-1}^T + (1 + Z_r^T V_{z(r-1)}^{-1} Z_r)^{-1} V_{z(r-1)}^{-1} Z_r (X_r - \hat{\gamma}_{r-1} Z_r)^T.$$

3 Optimal Contextual Support

The selection of an appropriate model support (I_r^c) is important to obtain good restoration results. If the contextual neighbourhood is too small it can not capture all details of the random field. Inclusion of the unnecessary neighbours on the other hand add to the computational burden and can potentially degrade the performance of the model as an additional source of noise. The optimal Bayesian decision rule for minimizing the average probability of decision error chooses the maximum posterior probability model, i.e., a model M_i corresponding to

$\max_j \{p(M_j|X^{(r-1)})\}$. If we assume uniform prior for all tested support sets (models) the solution can be found analytically. The most probable model given past data is the model M_i ($I_{r,i}^c$) for which $i = \arg \max_j \{D_j\}$.

$$D_j = -\frac{1}{2} \ln |V_{z(r-1)}| - \frac{\beta(r) - d\eta + 2}{2} \ln |\lambda_{(r-1)}| + \frac{d\eta}{2} \ln \pi \\ \left[\ln \Gamma\left(\frac{\beta(r) - d\eta + 2}{2}\right) - \ln \Gamma\left(\frac{\beta(0) - d\eta + 2}{2}\right) \right] . \quad (23)$$

4 Global Estimation of the Point-Spread Function

Similarly with (11) the degradation model (1) can be expressed in the matrix form

$$X_{r,i} = \psi W_r + \epsilon_{r,i} , \quad (24)$$

where $\nu = \text{card}(I_r)$,

$$\psi = [H_{1,i}, \dots, H_{\nu,i}] , \quad (25)$$

and W_r is a corresponding vector of Y_{r-s} . The unobservable $\nu \times 1$ image data vector W_r is approximated using (3), (8),(22), i.e.,

$$\hat{W}_r = [\hat{\gamma}_{r-s-1} Z_r]^T_{s \in I_r} . \quad (26)$$

In contrast to the model (10) the degradation model (1) is non-causal and hence it has no simple analytical Bayesian parameter estimate. Instead we use the least square estimate

$$\hat{\psi} = \min_{\psi} \left\{ \sum_{\forall r \in I} (X_r - \psi_r \hat{W}_r)^T (X_r - \psi_r \hat{W}_r) \right\} . \quad (27)$$

The optimal estimate is $\hat{\psi}^T = V_{\hat{W}}^{-1} V_{\hat{W}X}$ where the data gathering matrices $V_{\hat{W}}$, $V_{\hat{W}X}$ are corresponding analogies with the matrices (18),(19).

If we assume a non-homogeneous slowly changing point-spread function, we can estimate its local value using the local least square estimate analogously with (27) if the sum is restricted to a subwindow $J_r \subset I$.

5 Results

The four test sunspot images of the Fig.1, quantized at 256 levels per spectral band, were corrupted by the unknown PSF and the noise due to observation conditions. They are used as the input set for our algorithm.

In Fig.2 (left) is the reconstructed image using our method. This result we can compare with an original image observed in very good atmospheric conditions (right). Visual comparison of the reconstructed image with the input degraded images demonstrates clear deblurring effect of the presented algorithm. It is necessary to remark, that the "relatively" good original image in Fig.2 (right) was not used as an input image for the reconstruction method.

The performance of the presented method is compared with a restored image using the multichannel blind deconvolution method based on so-called subspace technique [24] in

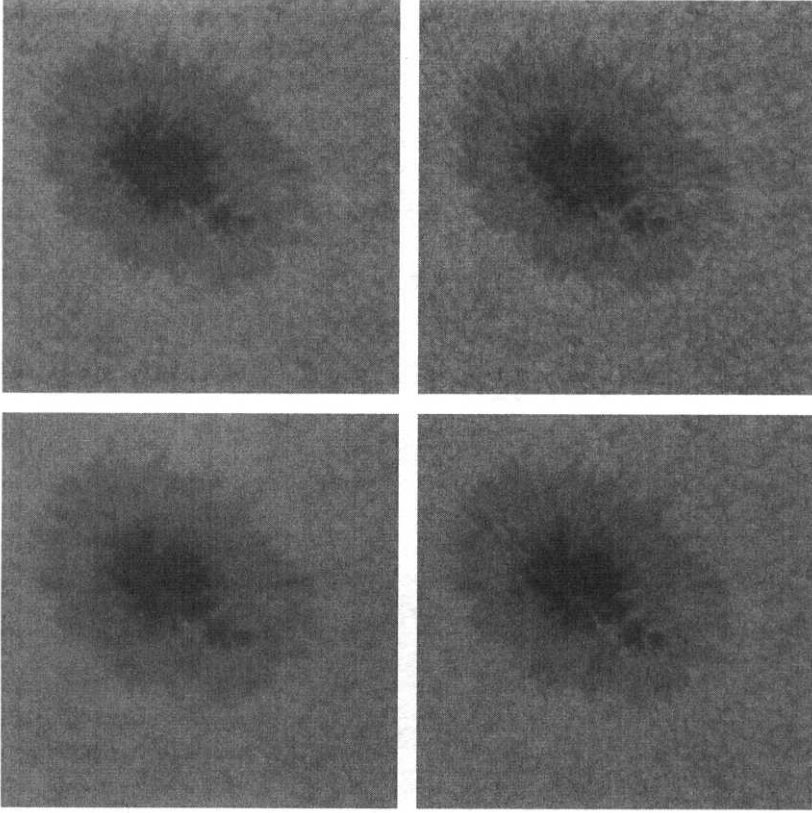


Figure 1: Examples of the four test images degraded by a varying atmospheric turbulence.

Fig.3. As an objective measure of the restoration performance an integral of a sum of image partial derivatives (28) has been used.

$$D(\hat{Y}) = \int \int \left| \frac{\partial \hat{Y}}{\partial r_1} \right| + \left| \frac{\partial \hat{Y}}{\partial r_2} \right| dr_1 dr_2 \quad (28)$$

If the unknown point-spread function H is non-negative, i.e., $H_s \geq 0 \quad \forall s$ and it preserves the image energy ($\int \int H_r dr_1 dr_2 = 1$) then $D(Y * H) \leq D(Y)$. This means that for the less blurred images the D is growing up.

Both proposed methods are superior over the classical methods using both criteria (28) and the visual one.

The Tab. 1 demonstrates the improvement of the deblurring and noise removal effect of the presented algorithm over the previously published ones. The proposed method is clearly superior for the degraded images.

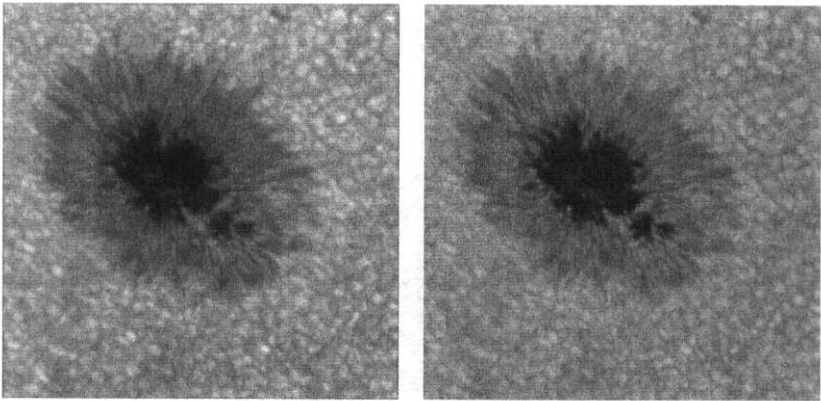


Figure 2: The reconstructed sunspot image using our method (left) and the original image observed in good atmospheric conditions (right).

Table 1: The measure of the restoration quality D evaluated for blurred and restored images.

Sunspot image restoration			
Image	Restored / Unrestored		D
Fig.1 upper left	U		5.80
Fig.1 upper right	U		4.80
Fig.1 lower left	U		4.24
Fig.1 lower right	U		4.55
Fig.3 left	R		15.15
Fig.3 right	R		13.41

6 Conclusions

The proposed recursive blur minimizing reconstruction method is very fast (approximately five times faster than the median filter) robust and its reconstruction results surpasses some standard reconstruction methods. Causal models such as (10) have obvious advantage to have the analytical solution for parameter estimation, prediction, or model identification tasks. However, this type of models may introduce some artifacts in restored images. These undesirable effects are diminished by introducing adaptivity into the model. This novel formulation allow us to obtain extremely fast adaptive restoration and / or local or global point-spread function estimation which can be easily parallelized. The method can be also easily and naturally generalized for multispectral (e.g. colour, multispectral satellite images) or registered images which is seldom the case for alternative methods. Finally, this method enables to estimate homogeneous or slowly changing non-homogeneous degradation point-spread function.

Acknowledgments

This research was supported by the GAČR grants no. 102/00/0030 and 102/00/1711 and partially supported by the GAČR grant no. 106/00/1715.

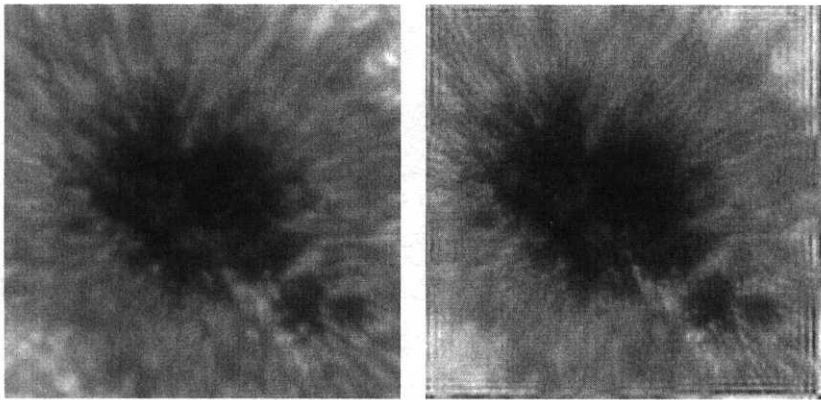


Figure 3: The detail part of reconstructed image using our method (left) and the algorithm [24] (right).

References

- [1] H. C. Andrews and B. Hunt, *Digital Image Restoration*. Prentice-Hall, Englewood Cliffs, 1977.
- [2] B. Chalmoud, Image restoration using an estimated markov model, *Signal Processing* **15** (1988) 115–129
- [3] R. Chellappa and R. Kashyap, Digital image restoration using spatial interaction models, *IEEE Trans. Acoustics, Speech and Sig. Proc.* **30** (1982) 284–295
- [4] K. Deguchi and I. Morishita, Two-dimensional auto-regressive model for the representation of random image fields, In: *Proc.ICPR Conf.*, IEEE, Munich (1982) 90–93
- [5] B. Fischl and E. Schwartz, Learning an integral equation approximation to nonlinear anisotropic diffusion in image processing, *IEEE Trans. Pattern Anal. Mach. Int.* **19** (1997) 342–352
- [6] D.L. Fried, *J. Opt. Soc. Am.*, **56**, 1372, 1966.
- [7] D. Geman, *Random fields and inverse problems in imaging*, Springer, Berlin, 1990.
- [8] S. Geman and D. Geman, Stochastic relaxation, gibbs distributions and bayesian restoration of images, *IEEE Trans. Pattern Anal. Mach. Int.* **6** (1984) 721–741
- [9] M. Haindl, Recursive model-based image restoration, In: *Proc. of the 13th ICPR Conf. vol. III*, IEEE Press, Barcelona (2000) 346–349
- [10] M. Haindl, Recursive square-root filters, In: *Proc. of the 13th ICPR Conf. vol. II*, IEEE Press, Barcelona (2000) 1018–1021
- [11] M. Haindl and S. Šimberová, A high - resolution radiospectrograph image reconstruction method, *Astronomy and Astrophysics, Suppl.Ser.* **115** (1996) 189–193
- [12] M. Haindl and S. Šimberová, A scratch removal method, *Kybernetika* **34** (1998) 423–428
- [13] M. Haindl, Recursive model-based colour image restoration,” In: *Structural, Syntactic, and Statistical Pattern Recognition*. Terry Caelli, Adnan Amin, Robert P.W. Duin, Mohamed Kamel, Dick de Ridder Eds., *Lecture Notes in Computer Science* 2396, ISBN 3-540-44011-9, Springer-Verlag, Berlin (2002) pp. 617 – 626
- [14] B. Hunt, The application of constraint least square estimation to image restoration by digital computer, *IEEE Trans. Computers* **22** (1973) 805–812
- [15] B. Jeffs and W. Pun, Simple shape parameter estimation from blurred observations for a generalized Gaussian mrf image prior used in map restoration, In: *Proc. IEEE CVPR Conf.*, IEEE, San Francisco (1996) 465–468

- [16] R. Lagendijk, J. Biemond and D. Boeke, Identification and restoration of noisy blurred images using the expectation-maximization algorithm, *IEEE Trans. on Acoustics, Speech, Signal Processing* **38** (1990) 1180–1191
- [17] J. Marroquin and T. Poggio, Probabilistic solution of ill-posed problems in computational vision, *J. Am. Stat. Assoc.* **82** (1987) 76–89
- [18] M. Nitzberg and T. Shiota, Nonlinear image filtering with edge and corner enhancement, *IEEE Trans. Pattern Anal. Mach. Int.* **16** (1992) 826–833
- [19] P. Perona, Deformable kernels for early vision, *IEEE Trans. Pattern Anal. Mach. Int.* **17** (1995) 488–489
- [20] P. Perona and J. Malik, Scale-space and edge detection using anisotropic diffusion, *IEEE Trans. Pattern Anal. Mach. Int.* **12** (1990) 629–639
- [21] S. Reeves and R. Mersereau, Identification by the method of generalized cross-validation, *IEEE Trans. Im. Processing* **1** (1992) 301–311
- [22] F. Roddier, *Progress in Optics*, XIX, ed. E. Wolf, Nort-Holland, 1981.
- [23] T. Schulz, *J. Opt. Soc. Am. A*, **10**(5), 1064, 1993.
- [24] F. Šroubek, J. Flusser, T. Suk and S. Šimberová, Multichannel Blind Deconvolution of the Short-Exposure Astronomical Images, in: *Proc. 15th IAPR Intl. Conf. Pattern Recognition '00*, **3**, pp. 53–56, Barcelona, Spain, 2000

Using a Helper FFN to Represent the Cost Function for Training DRNN's by Gradient Descent.

Roelof K. BROUWER
Department of Computing Science
University College of the Cariboo
Canada
Fax 250-371-5750
Telephone 250 374 5874
Email rkbrouwer@ieee.org

Abstract. This research is concerned with a gradient descent training algorithm for a target network that makes use of a *helper* feed-forward network (FFN) to represent the performance function used in training the target network. A *helper* FFN (HFFN) is trained because the mathematical form of the performance function for the target network in terms of the trainable parameters, \mathbf{P} , is not known yet data for the relationship can be generated. The transfer function of the HFFN provides a differentiable function for the performance function of the parameter vector, \mathbf{P} , for the target network allowing gradient search methods for finding the optimum \mathbf{P} . The method is applied to the training of **discrete** recurrent networks (DRNNs) that are used as a tool for classification of temporal sequences of characters.

1 Introduction

This research is concerned with a training algorithm based on gradient descent that makes use of a *helper* feed-forward network (FFN) to represent the performance function used in training a target network. The *helper* FFN is trained because the mathematical form of the performance function for the target network in terms of the trainable parameters is not available.

Values for the parameter vector, \mathbf{P} , of the target network are generated randomly and performance values are determined to produce the training data for the HFFN. This data is then used to train a HFFN with its own connection matrices. The input to this FFN is \mathbf{P} and the output is performance measure. The transfer function of the HFFN provides a differentiable function for the performance function of the parameter vector, \mathbf{P} , for the target network allowing gradient search methods for finding the optimum \mathbf{P} . The transfer function of the HFFN is the differentiable function that can be utilized to determine the correct \mathbf{P} for the target network.

The method is applied to the training of **discrete** recurrent networks (DRNNs) that are used as a tool for classification of temporal sequences of characters from some alphabet and identification of a finite state machine (FSM) that may have produced all the sequences. Classification of sequences input to the DRNN is based on the state that the network is in after the last element in the input sequence has been processed. At minimum then if the network is to be used for classifying sequences the terminal states for class 0 sequences

must be distinct from terminal states for class 1 sequences. The performance measure to be used in training must therefore be a function of this disjointedness and no more. The outcome of this is a performance relationship with no known mathematical representation.

The transform function of the HFFN is a differentiable function that can be used in the training of the DRNN using gradient descent. This method provides much greater flexibility in choice of performance measures and therefore also in the way the target network is used for classifying.

The rest of the paper is organized as follows. First is a description of the architecture of the network to which the proposed training method is applied. This is followed by a description of the training method including a definition of the objective or performance function. Next is the description of the simulations which is then followed by the summary.

2 The architecture of the DRNN to be trained using a HFFN

The DRNN has a finite and discrete set of states and has no explicit output except for state. The successor state, s' , is a deterministic function of the predecessor state, s , and an input vector u . Both s and s' are elements of S and u is an element of U where S and U are both finite sets. Both s and u are vectors whose components are assumed to be bounded. Elements of S have m components and elements of U have n components. The transition function for the DRNN is represented by (1)

$$s' = T(s, u) \quad (1)$$

There are two characteristics of representation of state, s , for the DRNN. One is the number of components in the state vector and the other is the granularity of its components defined as the number of possible values for a component. The number of components affects the number of learnable parameters in addition to determining the number of states while the granularity does not affect the number of learnable parameters.

Let N be the operator with the domain and range as shown in (2).

$$\mathcal{R}^m \xrightarrow{N} [-1, 1]^m \quad (2)$$

Specifically we let N correspond to normalization performed by dividing its argument by the maximum absolute value. The effect of N is to squash the components of its vector argument into the interval $[-1, 1]$. Let D , for discrete, be the operator with the domain and range as shown in (3) providing a granularity of $2g+1$. g is a natural number.

$$[-1, 1] \xrightarrow{D} \{-1, (-g+1)/g, \dots, (g-1)/g, 1\} \subset [-1, 1] \quad g \in \mathbb{N} \quad (3)$$

We may then write (1) as (4)

$$s' = D(N(f(s, u))) \quad (4)$$

D is defined according to (5) where the brackets indicate the floor function

$$\frac{\lfloor 0.5 + g \cdot x \rfloor}{g} \quad (5)$$

This is a generalization of rounding. In the case of rounding g is a power of 10. The result is a finite state space. Other possibilities for squashing exist such as the use of the sigmoid or tanh function. However these functions require the network to operate at saturation levels that cause problems during training if a FSM is to be modeled. Function f is defined by

$$f(s, u)_k = \sum_p W_{kp}^{(1)} * s_p + \sum_q W_{kq}^{(2)} * u_q + \sum_{i,j} W_{kij}^{(0)} * s_i * u_j \quad (6)$$

The latter corresponds to a second order recurrent network. It is second order because each component of the successor state is a linear combination of all products of state

components and input components. High-order networks are expansions of the first order Hopfield [3] and Cohen-Grossberg [2] models that allow high-order interactions between neurons since components of \mathbf{s} and \mathbf{u} are multiplied. The result may be written without subscripts in the directly executable mathematical notation, J [4].

$$\mathbf{f}(\mathbf{s}, \mathbf{u}) = (+/\mathbf{W}^{(1)} * \mathbf{s}, \mathbf{u}) + (+/ +/) "2 \mathbf{W}^{(0)} * \mathbf{s} */ \mathbf{u} \quad (7)$$

\mathbf{s}, \mathbf{u} is the catenation of \mathbf{s} and \mathbf{u} . $(+/\mathbf{W}^{(1)} * \mathbf{s}, \mathbf{u})$ means double summation over each of the 2-dimensional arrays following. $\mathbf{s} */ \mathbf{u}$ means vector cross product. $\mathbf{W}^{(1)}$ is a different array of course than before. Expression (7) has $m*(n+m)+m*m*n$ parameters. In our case we will consider (7) also without the first term as in (8)

$$\mathbf{f}(\mathbf{s}, \mathbf{u}) = (+/ +/) "2 \mathbf{W}^{(0)} * \mathbf{s} */ \mathbf{u} \quad (8)$$

The coding of input that will be used here is 1-of- n where n is the number of characters in the alphabet. If the alphabet is {a, b, c} then 'a' is represented by 100, 'b' by 010 and 'c' by 001.

The transition function for a specific finite state machine (FSM) may be represented by a table with rows corresponding to states and columns corresponding to input values. Entries in the table are the successor states for states. An example of the transition table for a FSM is Table 1. The states are represented by non-negative integers rather than by n -tuples as in the case of DRNNs. This is possible because the state identifiers do not have to play a role in calculations. The input alphabet also consists of single characters here rather than m -tuples. The tuples apply in the case of the network representing the FSM.

Table 1 Example of a state transition table for a FSM

Input→ State ↓	a	b	c
0	2	3	1
1	3	2	1
2	1	2	3
3	2	3	1

3 Training method for the DRNN

3.1 Introduction

Since learning by recurrent neural networks to generate or classify temporal patterns, or to memorize oscillatory patterns may be viewed as an optimization problem most numerical methods of optimization, like back-propagation [5], are based on methods of steepest descent that compute the gradient to permit taking a step in the direction of maximum improvement.. However learning long term dependencies is difficult [1].

In the method suggested here a function that approximates the relationship is determined by training a FFN. As long as fit values or performance measure values can be determined for a fixed set of training data for various potential values of the parameter vector then this data can be used to train a FFN. The sought after mathematical relationship may then be approximated by the transfer function of the FFN. The gradient used in training the target network is simply the gradient of the transfer function of the HFFN. This will always have

the same form regardless of the target network. This gradient will not involve any recursion even though the target network is recurrent.

3.2 Objective functions or performance measures for training

Let the two classes of sequences to be separated be called class 0 and class 1. The method by which the network is to be used for classification and the performance measure should be as non-constraining as possible and yet lead to a network which can after training be used for correct classification of new sequences. Classification is then done according to the terminal state of an input sequence. It must therefore be possible to separate the terminal states for the class 0 training sequences from the terminal states for the class 1 training sequences.

Let F represent the mapping from a sequence to a state s where s is the state the network is in after all components of the sequence have been processed.

$$F(s_0, a_0 a_1 \dots a_{k-1}) \triangleq T(\dots T(T(s_0, a_0), a_1), \dots a_{k-1}) \quad (9)$$

s_0 is the initial state. Let G represent the mapping from sets of sequences to sets of states based on the mapping F from a sequence to a state.

$$s \in G(s_0, \text{Seq}) \Leftrightarrow (s = F(s_0, \text{seq}) \wedge \text{seq} \in \text{Seq}) \quad (10)$$

Seq with capital S is a set of sequences and seq is a particular sequence. Let ts_0 be the set of class 0 training sequences and ts_1 the set of class 1 training sequences. Let es_0 be the set of terminal states for training sequences of class 0 and es_1 the set of terminal or terminal states for training sequences of class 1. Then we can write

$$es_0 \triangleq G(s_0, ts_0) \quad (11)$$

$$es_1 \triangleq G(s_0, ts_1) \quad (12)$$

G in general is many-to-one with many sequences mapping to the same terminal state. However for classification based on terminal states to work correctly $es_0 \cap es_1$ should be the null set. Elements in $es_0 \cap es_1$ are ambiguous terminal states since each image is the terminal state for both a class 0 and a class 1 sequence.

Let ts_a be the set of sequences that map to ambiguous states as in (13).

$$ts_a \text{ is such that } G(s_0, ts_a) = (es_0 \cap es_1) \quad (13)$$

We will use the proportion of training sequences that map to ambiguous terminal states as part of a performance measure. Now

$$\#ts_a = 0 \Leftrightarrow \#(es_0 \cap es_1) = 0 \quad (14)$$

That is if $\#(es_0 \cap es_1) = 0$ or $\#ts_a = 0$ during training then each training sequence maps to the correct terminal state and we may say that the training sequences are 100% correctly classified. The only concern is that a labeling was selected that would produce the correct classification. This is obviously somewhat biased. Let us therefore consider an extension to this.

For the complete performance measure the training sequence set is partitioned into two so that we obtain four sets in total. Let the labels for the sets be ts_0 , ts_1 , $vs_{1,0}$ and vs_1 . t represents training and v represents validation. During training ts_0 and ts_1 are used to determine the terminal states for class 0 and class 1 respectively as before since they are not predefined and only produced while the training takes place. During training vs_0 and vs_1 are then used to determine if classification of new sequences can be performed adequately using the terminal states defined by ts_0 and ts_1 . The sets of sequences, vs_0 and vs_1 , are like

validation sets used in back-propagation sometimes except that they play more of a key role here than in the case of back-propagation.

The set of terminal states corresponding to class 0 validation sequences should be a subset of the class 0 terminal states for class 0 training sequences and similarly for class 1. Since the class of an input sequence is the class of its terminal state the class of a terminal state that a sequence maps to should be unambiguous. The goal is that the images (terminal states) of ts_0 and ts_1 are disjoint and that the images of vs_0 and vs_1 are subsets of es_0 and es_1 respectively.

The complete performance measure or fit value that is given by (15)

$$r \frac{\#ts_a}{\#ts_0 + \#ts_1} + (1-r) \left(1 - \frac{\#vs_0' + \#vs_1'}{\#vs_0 + \#vs_1}\right) \quad (15)$$

With vs_0' and vs_1' defined by

$$\text{seq} \in vs_0' \Leftrightarrow \mathbf{F}(s_0, \text{seq}) \in es_0 \quad (16)$$

$$\text{seq} \in vs_1' \Leftrightarrow \mathbf{F}(s_0, \text{seq}) \in es_1 \quad (17)$$

$\#vs_0'$ and $\#vs_1'$ are the number of validation sequences in class 0 and class 1 respectively that map to training terminal states for class 0 and class 1 respectively. r is a number between zero and one and is used to control the relative importance of two measures.

3.3 The HFFN

Assume that we can determine performance values for values of a parameter vector based on a set of training data for a target network to be learned. We then have the data for a relationship between performance value, Q , and parameter vector, \mathbf{P} , for a target network to be trained. We may approximate this relationship by a HFFN. The relationship becomes the training and test data for the HFFN. The transform function, (18), corresponding to the HFFN can then be used in training the target network using gradient descent.

$$Q = \text{sigm } \mathbf{W}^{(1)} \cdot \text{sigm}(\mathbf{W}^{(0)} \cdot \mathbf{P}) \quad (18)$$

\mathbf{P} is the parameter vector for the target network to be trained. Here $\mathbf{W}^{(0)}$ and $\mathbf{W}^{(1)}$ are the connection matrices for the first and second layer respectively of the HFFN and are not the same as for the target network. The gradient of this performance measure with respect to \mathbf{P} , (19), will then be used in training the target using gradient descent.

$$\nabla_{\mathbf{P}} Q = \text{sigm}'(\mathbf{W}^{(1)} \cdot \text{sigm}(\mathbf{W}^{(0)} \cdot \mathbf{P})) * (\mathbf{W}^{(1)} * \text{sigm}'(\mathbf{W}^{(0)} \cdot \mathbf{P})) \cdot \mathbf{W}^{(0)} \quad (19)$$

$\mathbf{W}^{(1)}$ and $\text{sigm}'(\mathbf{W}^{(0)} \cdot \mathbf{v})$ are row vectors and so is the product $\mathbf{W}^{(1)} * \text{sigm}'(\mathbf{W}^{(0)} \cdot \mathbf{P})$. Note that there is no back-propagation through layers or through time in the case of the training of the target DRNN. The only back-propagation is that which is involved in the training of the HFFN.

4 Simulations

Several simulations were run to determine the effectiveness of the proposed method. First the method was applied to the training of a DRNN based on (8). Figure 1 shows the result of a training session in terms of fit values versus number of training epochs in 25ths. The training and test sequences were generated on the basis of transition table Table 2

Table 2 Transition table used to generate training and test sequences. Terminal states are 4 for class 0 and 6 for class 1

input→ ↓state	a	b	c
0	9	1	2
1	2	7	8
2	9	0	8
3	3	8	4
4	6	5	1
5	1	0	8
6	9	3	4
7	0	5	7
8	6	4	1
9	2	6	1

The final states are 6 and 4 for class 0 and class 1 respectively. The percent correct after training was 100 % for class 0 and 97.5 for class 1. 1000 epochs were used to train the HFFN to represent the performance function although stability was reached after 850 epochs. 63 training values were used to train the HFFN. Two components were used for state, the granularity was 41 and the number of components in the input was 3. The values of $\mathbf{W}^{(0)}$, the sought after parameter matrix for the target network, were restricted to lie in $\{-1,0,1\}^{2^{22} \times 3}$ for the purpose of generating the training data for the HFFN. Two hundred strings were generated for each class, 80% of which were used for training. Of the training data 20% were used to define the classification of terminal states and 80% were used for validation to determine an overall fit value.

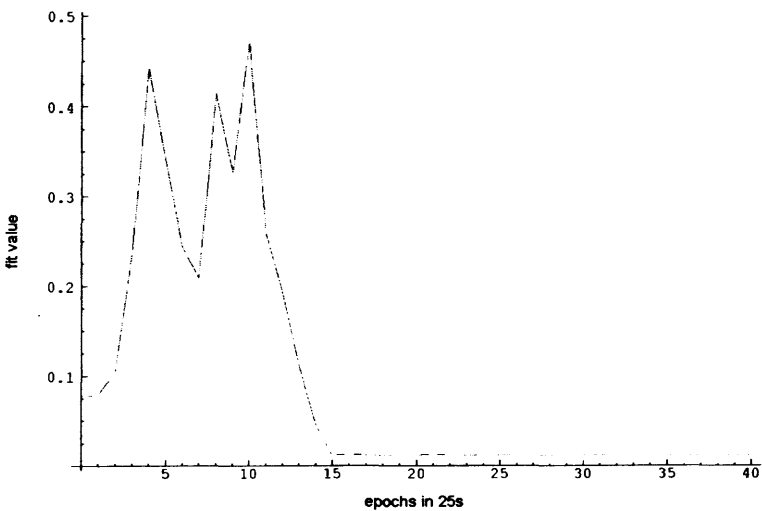


Figure 1 Performance values versus number of epochs in 25ths

The graph shows that the approach to the optimal performance value is not smooth. This is expected to be due to the lack of smoothness in the fit relationship. However 100% accuracy on the test data was achieved after less than 200 epochs.

The next simulation corresponds to the training of a DRNN based on (7) using the same set of training and test sequences and other parameters. The performance during training is shown in Figure 2. The graph again demonstrates an erratic approach to a fit value of 0.

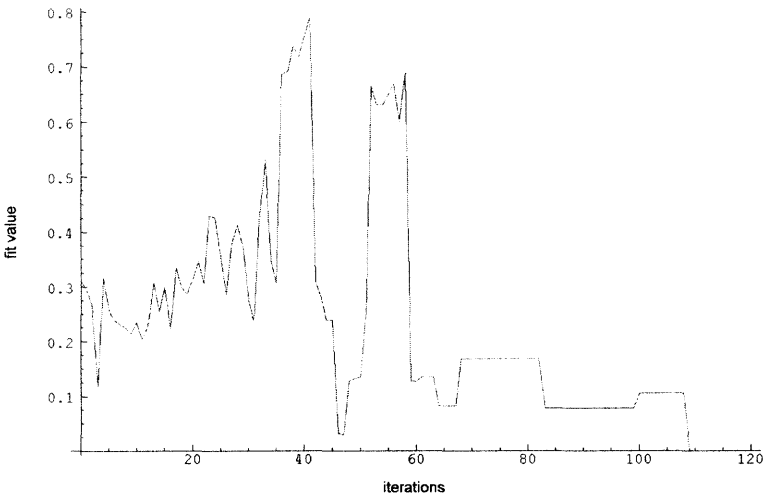


Figure 2 Performance values versus number of epochs using extended network

In this case 100 % correctness on the test data was achieved after only 110 epochs. Note that in this case there are a much larger number of parameters.

It should be noted that 100% correct is not always achieved after 1000 epochs and depends on the sets of training sequences. Following are results for randomly produced transition tables. Percent correct results for several runs with the extended network of (7) are shown in Table 3.

Table 3 Value of percent correct after 1000 epochs

27.5	22.5	41.25	100	56.25	61.25	18.75	100	35	100
------	------	-------	-----	-------	-------	-------	-----	----	-----

Training was only allowed to go up to 1000 epochs because of time constraint.

5 Summary

This research is concerned with a training algorithm based on gradient descent that makes use of a *helper* feed-forward network to represent the performance function used in training a target network. The HFFN is trained because the mathematical form of the performance function for the target network in terms of the trainable parameters is not known yet data for the function can be generated.

The gradient of the transition function of the HFFN is then used in the gradient descent method for training a target network. The form of this gradient is not dependent upon the target network although its parameters are. Even though the target network to be trained is recurrent the gradient used in training will not be recursive with its own problems of passing errors through much iteration. This method also allows great flexibility in the choice of fit function or performance measure. However if the relationship between fit

value and parameters of the target vector is not very smooth the transition function of the HFFN will not be very smooth either and training will still be difficult. Therefore further work is still required in that area. Additional work could also focus on the graph associated with the transition table to see why in some cases the equivalent DRNN is found more readily than in others.

6 References

- [1] Y. Bengio, P. Simard, and P. Frasconi. "Learning long term dependencies with gradient descent is difficult." *IEEE Transactions Neural Networks*, 5:147-166, 1994.
- [2] M. A. Cohonen and S. Grossberg. "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks". *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, pp. 815-826, 1983.
- [3] J. J. Hopfield "Neural networks and physical systems with emergent collective computational abilities" *Proceedings of the national academy of science. USA* vol. 79, pp. 2554-2558.
- [4] K. E. Iverson 1996, *J Introduction and Dictionary*. Iverson Software Incorporated
- [5] P. J. Werbos "Back propagation through time: what it does and how to do it", *Proceedings IEEE*, Vol. 78, pp. 1550-1560.

7 Appendix

7.1 Nomenclature

- Vectors are always assumed to be column vectors unless otherwise specified, and they are represented by lowercase boldface letters such as **a**, **b**, and **x**.
- Matrices are represented by bold capital letters, such as **A**, **B**, and **M**.
- **0** is the vector consisting of all 0's.
- **1** is the vector consisting of all 1's.
- An extension of function **f** applied to a vector **y** = (**y**₀, **y**₁, -- **y**_{*n*-1}), **f**(**y**), is defined as **f**(**y**) = (**f**(**y**₀), **f**(**y**₁), -- **f**(**y**_{*n*-1})).

Following is a description of the terms and operations used in this document, some of which may be new. The newness of some of the notation used in this paper is far outweighed by the conciseness that the notation introduces. The functions are from the **J** notational system [Iver96] for manipulating arrays that permits whole arrays to be manipulated. This means that subscripts may be avoided in several cases leading to less cluttered expressions.

- **x * y** is the same as **x** times **y**.
- **/** means over all. **+ / x** is the summation of all components in vector **x**.
- **(x * / y)** is the cross product. Therefore **(x*/y)_{i,j}** has the value **x_i * y_j**.
- **A op B** is defined as long as the dimensions of **A** form a prefix of the dimensions of **B** or if the dimensions of **B** form a suffix of the dimensions of **A**. **op** is a binary operator for two scalar arguments such as **+**, **-**, ***** etc. For example if **A** is 3-dimensional and **B** is 2 dimensional **(A op B)_{i,j,k} ≡ A_{i,j,k} op B_{j,k}**
- **op "2** means that operator **op** should be applied to the two dimensional sub arrays of its argument obtained by considering the last two dimensions.

Scene-Based Nonuniformity Correction Method Using the Inverse Covariance Form of the Kalman Filter

Sergio Torres I. (e-mail: storres@die.udec.cl), Jorge Pezoa N. (e-mail: jpezoa@die.udec.cl)
Department of Electrical Engineering, Universidad de Concepción, CHILE

Abstract A scene-based algorithm for nonuniformity correction (NUC) in focal-plane arrays (FPA) detectors has been developed. The NUC technique is based in the inverse covariance form (ICF) of the Kalman filter. The gain and the offset of each detector of the FPA are modeled by discrete-time Gauss-Markov processes. These parameters are taken as constant within a given sequence of frames, corresponding to a certain time and operational conditions, but they randomly drift from one sequence to another in response to new operational conditions. For each detector and each sequence of frames, the ICF filter input is an observation vector consisting of detector's read-out values. The output of ICF filter for any sequence of infrared frames is the detectors' gain and offset. The efficacy of the ICF of the Kalman filter to compensate for nonuniformity noise in infrared imagery is demonstrated using sequences of infrared imagery with both artificial nonuniformity and artificial drift in the detectors' parameters. It is shown that the ICF filter and the Kalman filter generate similar reductions of nonuniformity. However, the ICF filter compensates the noisy images with less number of operations per pixel and per frame than the Kalman filter.

1 Introduction

Focal-plane array (FPA) detectors are used in the visible and infrared spectral bands imaging systems for a variety of applications. A FPA consist of a two-dimensional array of photodetectors placed in the focal plane of the imaging lens. The performance of a FPA is strongly affected by the nonuniformity response of the detectors.

The problem of nonuniformity in array detectors is attributed to the fact that each detector in the array has a different responsivity to light. Thus, for an imaging sensor viewing a spatially constant scene, the sensor output may not be constant from pixel to pixel (as one would hope). This nonuniformity is also referred to as fixed-pattern noise and it may severely corrupt the images. In severe cases, the result is like looking at the world through a "dirty window". As one might expect, the fixed-pattern noise reduces the resolving capability of an imaging system. For example, it may not be possible to capture minute pixel-to-pixel temperature variations. In addition, the fixed-pattern noise causes the performance degradation of many frequently used post-processing operations which are typically employed to identify object motion. What makes matters worse is that the fixed-pattern noise slowly changes over time (in the course of minutes to hours in some cases) preventing a simple one-time laboratory calibration. Of course, one can calibrate frequently; however, this requires sophisticated and expensive calibration targets (i.e., black-body radiation sources) and more important, requires

halting normal imaging operations during the calibration process. In contrast to calibration-based nonuniformity correction (NUC), scene-based NUC is the process of compensation for the nonuniformity by using only the scenes frames being imaged.

Hayat's group [1–9] has been actively pursuing the development of novel algorithms for NUC based on statistical estimation theory. In particular, they developed a Gauss-Markov model for the slow variation in the fixed-pattern noise and utilized the model to estimate the nonuniformity parameters using a Kalman filter. Generally speaking, Kalman filters are computationally efficient estimation techniques that facilitate the calculation of a current estimate from the past estimate and current data. In the context of array detectors, blocks of frames (each containing hundreds of frames) separated by long time intervals (several minutes, in practice) are used to estimate the nonuniformity parameters. The nonuniformity parameters slightly vary from block to block. As new blocks arrive, the nonuniformity parameters corresponding to each new block are estimated by updating the old parameters (corresponding to the old block of frames). In this way, the valuable information contained in the old estimates are preserved and efficiently used in forming the current state of nonuniformity.

In this paper, we propose a scene-based NUC method which optimally estimate the gain and the offset of each detector in the FPA by means of the inverse covariance form (ICF) of the Kalman filter. The ICF of the Kalman filter provides an algorithm that is both mathematically equivalent to the conventional Kalman filter algorithm and computationally more efficient, when the dimension of the measurements is much greater than the dimension of the process noise. The matrix inverse operation is a very time consuming operation, and in the ICF filter we compute the inverse of a matrix with the dimensions of the noise process, while in the Kalman filter we compute the inverse of a matrix of the dimension of the observations. Further, the inverse covariance filter is better suited for problems where there exists no previous knowledge of the initial system state.

This paper is organized as follows. In section 2 the model of the system and the original Kalman filter are presented, and the ICF is developed following the standard procedures given in [12]. In section 3 the NUC technique is tested with sequences of infrared data with simulated nonuniformity and drift in time, and three performance parameters are computed to measure the performance of the method. In Section 4, we present the main conclusions.

2 Estimation of the Gain and Offset Using the Inverse Covariance Form of the Kalman Filter

When the FPA is operating in its linear input-output range, every detector may be modeled using an irradiance-linear model for the detector response, presented in [10] by Holst:

$$Y_{ij}(t) = x_{ij}^{(1)}(t) T_{ij}(t) + x_{ij}^{(2)}(t) + v_{ij}(t) \quad (1)$$

where the subscripts ij mean the position of the pixel in the array and t represents the temporal variation of the parameters in the model, $x_{ij}^{(1)}(t)$ and $x_{ij}^{(2)}(t)$ are ij -th the detector's gain and offset, respectively, at any time t . $T_{ij}(t)$ represents the ij -th input irradiance, at any time t , over the detector during integration time, $v_{ij}(t)$ is the ij -th additive readout noise and $Y_{ij}(t)$ is the ij -th readout signal for the detector response, both at any time t . From now on, the pixel subscripts ij will be omitted with the understanding that all operations are performed on a pixel-by-pixel basis.

2.1 The Kalman Filter

Torres *et al.*, [5] [7], developed a Kalman filter that estimates the gain and the offset of each detector in the FPA. To obtain the filter, the system states variables (gain and offset) were modeled using a Gauss-Markov random process. This model effectively captures the slow temporal variation inherent in the gain and the offset observed in many practical applications.

The state equations for the Gauss-Markov process at the k -th block time are given by:

$$\begin{bmatrix} X_{ij}^{(1)}(k+1) \\ X_{ij}^{(2)}(k+1) \end{bmatrix} = \begin{bmatrix} \alpha_k & 0 \\ 0 & \beta_k \end{bmatrix} \begin{bmatrix} X_{ij}^{(1)}(k) \\ X_{ij}^{(2)}(k) \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} W_{ij}^{(1)}(k) \\ W_{ij}^{(2)}(k) \end{bmatrix} \quad (2)$$

$$X_{k+1} = \Phi_k X_k + G_k W_k \quad (3)$$

again, $X^{(1)}(k)$ and $X^{(2)}(k)$ are detectors' gain and offset, $W^{(1)}(k)$ and $W^{(2)}(k)$ are the driving noise for the gain and the offset at the k th block. α_k and β_k parameters represent the amount of drift, in gain and offset, between k and $k+1$ block time. Values of α_k and β_k near to 1 generate slow drift and values near to 0 produce high drift in gain and offset.

Based on the state equation (3), on the extension of the observation equation (1) and assuming that for each frame in a block, the collected irradiance is a uniformly distributed random variable in some range $[T_{min}, T_{max}]$, common to all detectors and all time, the following Kalman Filter was derived [5, 7]:

$$\hat{X}_k^- = \Phi_{k-1} \hat{X}_{k-1} + M_{k-1}^T \quad (4)$$

$$P_k^- = \Phi_{k-1} P_{k-1} \Phi_{k-1}^T + G_{k-1} Q_{k-1} G_{k-1}^T \quad (5)$$

$$K_k = P_k^- \bar{H}_k^T [\bar{H}_k P_k^- \bar{H}_k^T + S_k]^{-1} \quad (6)$$

$$S_k = R_k + \sigma_T^2 (\sigma_{X_0^{(1)}}^2 + \bar{X}_0^{(1)}) I_{k,l_k} \quad (7)$$

$$\hat{X}_k = \hat{X}_k^- + K_k (Y_k - \bar{H}_k \hat{X}_k^-) \quad (8)$$

$$P_k = (I_{2,2} - K_k \bar{H}_k) P_k^- \quad (9)$$

with initial conditions:

$$\hat{X}_0 = E(X_0) = \begin{bmatrix} \bar{X}_0^{(1)} \\ \bar{X}_0^{(2)} \end{bmatrix}, \quad P_0 = \Lambda = \begin{bmatrix} \sigma_{X_0^{(1)}}^2 & 0 \\ 0 & \sigma_{X_0^{(2)}}^2 \end{bmatrix} \quad (10)$$

Equations (4) y (5) are the Kalman filter time updates, (6) is the Kalman gain and equations (8) and (9) are the measurement updates. \hat{X}_k^- is the a priori state estimate, \hat{X}_k is the current state estimate. P_k^- is the a priori error covariance matrix, P_k is the current error covariance matrix. K_k is the Kalman gain, Φ_{k-1} is the state transition matrix, Q_{k-1} and R_{k-1} are the auto and cross covariance functions of the driving noise and additive noise, respectively. G_{k-1} is the system associated matrix and Y_k is the vector of observations. \bar{H}_k is the mean of the observation matrix.

M_k is a vector containing the mean of the driver noise, σ_T^2 is the variance of the input infrared irradiance, $\bar{X}_0^{(1)}$ and $\sigma_{X_0^{(1)}}^2$ are the mean and variance of the initial condition of the gain, respectively. I_{k,l_k} is an identity matrix with dimensions of the block length. The matrixes R_{k-1} , S_{k-1} are a diagonal square matrix of dimension of the observation vector.

Table 1: Number of operations, per pixel and per block of frames, required for one iteration. n represents the dimension of the system state variables, m the dimension of the observations and p the dimension of the process noise

	Kalman Filter	Inverse Covariance Form
Adds	$3n^3 + n^2(2m + p - 2) +$ $+n(m - 1 + 3m^2 + p^2 + 2p) + m^3$	$4n^3 + n^2(4p - m - 1) +$ $+n(2p^2 - 3p - m + 2m^2 - 1) + m + m^2 + p^3$
Multiplies	$3n^3 + n^2(5m + p + 2) +$ $+n(4m + 2m^2 + p^2 + p) + m^3$	$4n^3 + n^2(4p + m + 3) +$ $+n(2p^2 + m + 2m^2) + 2m + p^3$

2.2 Derivation of the Inverse Covariance Form of the Kalman Filter

The ICF or information matrix form of the Kalman filter is an alternative method of the filter that is particularly well suited to problems where the measurement dimension is large or where there exists no a priori knowledge of the initial system state.

In NUC methods for FPAs, the number of observations ($\mathbf{1}_k$), per detector, normally are between 300 to 2000 frames [5, 7], so the dimension of the vector of observations is much greater than dimension of the noise and system states, the gain and offset in our case. Table 1 shows the number of operations required for the k th iteration for the Kalman Filter and for the ICF of the filter. Using a block length of 500 frames the Kalman filter needs 126754038 adds and 126014052 multiplies while inverse covariance filter calculates 1247570 adds and 1004100 multiplies.

We can see that the inverse covariance filter provides a computationally more efficient algorithm than the conventional Kalman filter. Further, for estimation problems where there exists no a priori knowledge of the initial system state, this algorithm is less susceptible to saturation problems within the equations and in this sense may provide a numerically more stable approach [12].

The ICF is an alternative form of the Kalman filter in which the inverse of the error covariance matrix, P_k^{-1} and the estimate:

$$\hat{\mathbf{a}}_k \triangleq P_k^{-1} \hat{\mathbf{X}}_k \quad (11)$$

are propagated in each iteration of the filter, rather than the error covariance matrix and the estimate $\hat{\mathbf{X}}_k$. The estimate of the system state, $\hat{\mathbf{X}}_k$, can be recovered from $\hat{\mathbf{a}}_k$ by multiplying $\hat{\mathbf{a}}_k$ by the matrix P_k .

The derivation of the filter was made following the standard procedure given in [12], consisting in the application of the matrix inversion lemmas: $(L + M N^T)^{-1} = L^{-1} - L^{-1} M (I + N^T L^{-1} M)^{-1} N^T L^{-1}$ and $(L + M N^T)^{-1} M = L^{-1} M (I + N^T L^{-1} M)^{-1}$ and the combination of the Kalman filter equations using the definition 11.

The equations for the ICF of the Kalman filter are:

$$\hat{\mathbf{a}}_k^- = \{I - B_{k-1} G_{k-1}^T\} \{\Phi_{k-1}^{-1} \hat{\mathbf{a}}_{k-1} + A_{k-1} M^T k_{k-1}^T\} \quad (12)$$

$$(P_k^-)^{-1} = (I - B_{k-1} G_{k-1}^T) A_{k-1} \quad (13)$$

$$A_{k-1} \triangleq \Phi_{k-1}^{-T} P_{k-1}^{-1} \Phi_{k-1}^{-1} \quad (14)$$

$$B_{k-1} \triangleq A_{k-1} G_{k-1} \{Q_{k-1}^{-1} + G_{k-1}^T A_{k-1} G_{k-1}\}^{-1} \quad (15)$$

$$P_k^{-1} = (P_k^-)^{-1} + \overline{H}_k^T S_k^{-1} \overline{H}_k \quad (16)$$

$$\hat{\mathbf{a}}_k = \hat{\mathbf{a}}_k^- + \overline{H}_k^T S_k^{-1} Y_k \quad (17)$$

The ICF of the Kalman filter is given by the the time updates (equations (12) and (13)), the measurement updates (equations (17) and (16)) and the initial conditions: $P_k^{-1} = \Lambda^{-1}$ and $\hat{a}_0 = \Lambda^{-1}\bar{X}_0$

In cases when there is no a priori knowledge of the initial system state, the inverse covariance filter can be utilized with the initial condition $P_k^{-1} = 0$, corresponding to a P_0 of infinity, situation that leads the traditional filter to saturation problems and consequently a significant and possibly catastrophic loss in numerical accuracy.

Furthermore, with the Kalman filter, we must invert the matrix $[\bar{H}_k P_k \bar{H}_k^T + S_k]^{-1}$, however with in the ICF the matrix to invert is $[S_k]^{-1}$, both have a dimension of the observation vector, but S_{k-1} is a diagonal matrix, so the computing time is considerably reduced.

3 Applications to Simulated Infrared Data

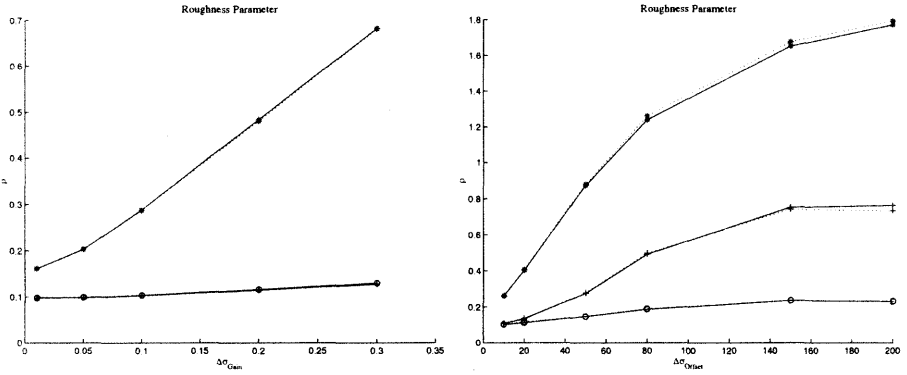


Figure 1: Roughness parameter for Kalman and ICF filter versus variations in the gain (left) and the offset (right) standard deviations, respectively. Solid (dotted) lines represents low (high) drift in parameters. On both pictures, the upper line is the roughness parameter of the uncorrected block. For the right graph, the middle lines are roughness obtained for ICF filter and lower lines roughness for Kalman filter.

The performance of the inverse covariance filter is studied and compared with the performance of the traditional Kalman filter using infrared image sequences that are corrupted by simulated nonuniformity. NUC is performed by simple subtracting the estimated offset from the data and dividing the outcome by the estimated gain. To study the performance, we use the mean-square error (MSE) for the estimated gain and offset, averaged over all detectors. NUC capability is examined in terms of the root-mean-square error (RMSE) and roughness parameter (ρ) for a corrected image, metrics commonly used in NUC [2, 10, 11]. All the assumptions for implementing the simulations are common for testing NUC methods and are given in [1–7, 11]. The inverse covariance filter assumed an unknown initial condition, $P_0^{-1} = 0$.

3.1 Performance on a Block of Infrared Frames

Figure (1) shows the computed roughness parameter, the upper line in both graphics is the roughness of the uncorrected image. It can be seen that, under variations in gain, even under

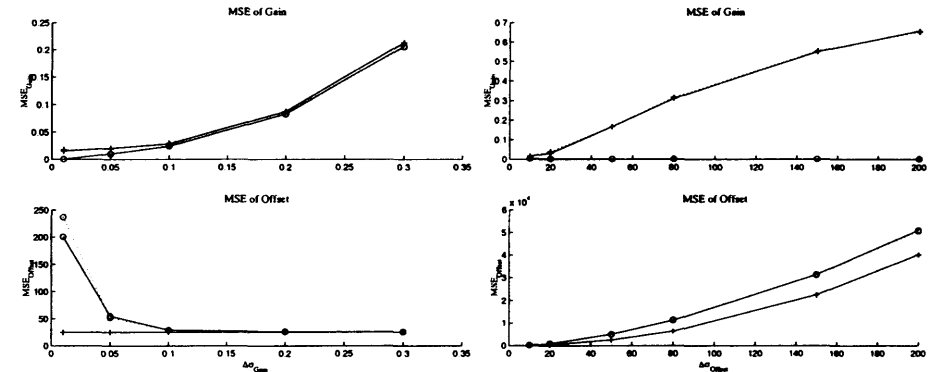


Figure 2: Computed MSE of the estimated gain (left) and offset (right) for the Kalman and ICF filter versus variations in the gain and offset standard deviations. Solid (dotted) lines represents low (high) drift in parameters, 'o' sign means Kalman filter and '+' sign means ICF filter

high drift in the parameters, there is a great reduction in the nonuniformity and it's similar for both filters. However, under offset's variations, the inverse covariance filter obtain a lower performance, situation that can be enhanced taking a larger number of frames. Figure (2) shows the mean-square error of the estimated gain and offset. Again, similar behavior for the estimation are obtained under variations in gain and offset, but we must note that, under offset's variations, the MSE of the gain is greater for the inverse covariance filter. However the offset's MSE is approximated the same for both filters. In Figure (3) we can see the computed root mean-square error for the corrected block of frames. Note that again the inverse covariance performance is less than traditional filter for the case with offset nonuniformity. As an example, figures (4) and (5) show the true image, the true image with simulated nonuniformity and the ICF corrected image with mainly gain nonuniformity and with mainly offset nonuniformity, respectively.

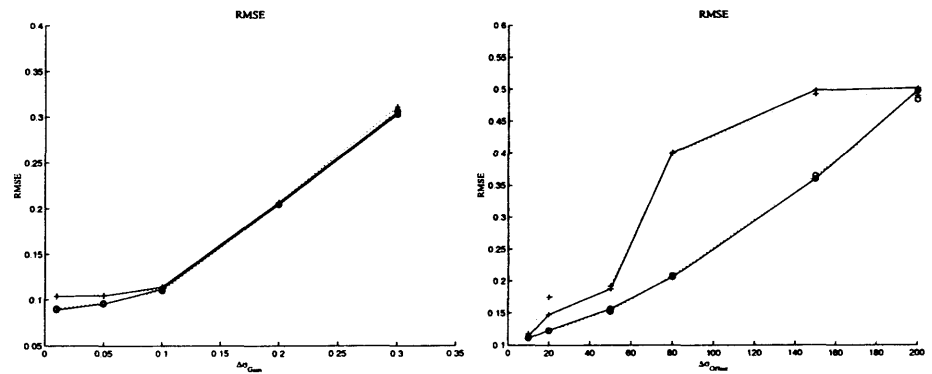


Figure 3: RMSE of the corrected image for traditional and inverse covariance filters under variations in the gain (left) and the offset (right). Solid (dotted) lines represents low (high) drift in parameters.

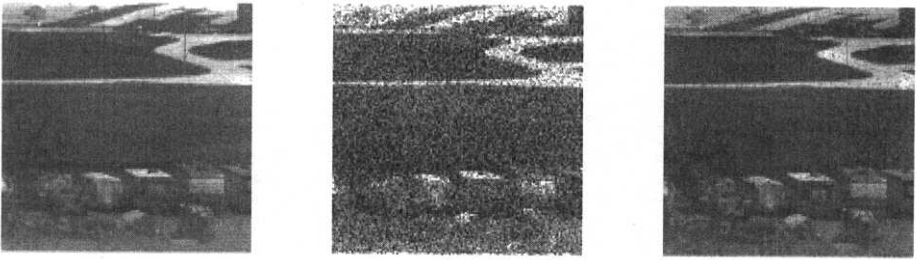


Figure 4: The True image (left), the True image with artificial nonuniformity (middle) and the ICF Corrected (right) frame. The nonuniformity is mainly generated for the gain.



Figure 5: The True image (left), the True image with artificial nonuniformity (middle) and the ICF Corrected (right) frame. The nonuniformity is mainly generated for the offset.

Figure (6) shows the CPU time consumed by the Kalman filter and ICF filter, versus the block length. By reducing the number of operations, the CPU time for the ICF filter has been reduced several times compared with the Kalman filter for any frame size and block lengths greater than 200 frames. For example, reductions of 55% in time is obtained for block lengths of 1000 frames. The tests were made using a Pentium IV 1.6 GHz processor and a 768 MBytes RAM and using the Matlab's function *cputime*.

3.2 Performance on a Set of Blocks of Infrared Frames

The ICF of the Kalman filter is evaluated using several blocks with simulated nonuniformity patterns at different levels of drift and each block with a fixed length of 500 frames. The results of the tests are presented in Table 2. The RMSE parameter is approximated the same for each k -th block, independent of the values of α and β (i.e., for high, moderated and low levels of drift). Further, the MSE of the estimated gain and offset prove that the ICF of the Kalman filter can effectively estimatimates the system parameters, independent of the values of α and β . This demonstrate that the ICF filter is capable of taking advantage of information contained in a previous block of frames and effectively updating this information using the current block.

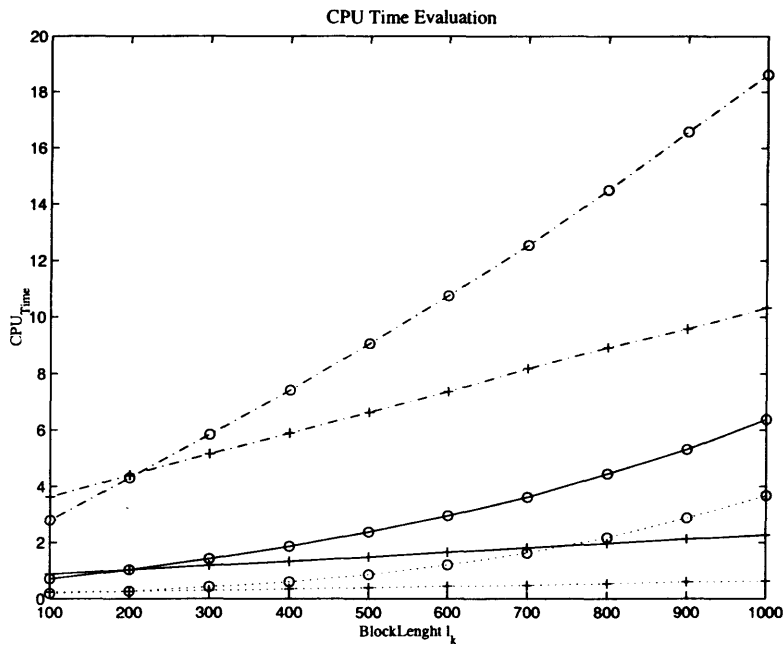


Figure 6: The CPU time consumed by the traditional Kalman and ICF filter for frames of 32x32 (dotted line), 64x64 (solid line) and 128x128 (dashdot line) pixels, 'o' sign means Kalman filter and '+' sign means ICF filter

Table 2: Performance of the inverse covariance filter estimating states on several blocks.

	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	k=10
$\alpha = \beta$	0.95	0.10	0.99	0.50	0.30	0.96	0.80	0.40	0.20	0.30
$MSE\hat{X}^{(1)}$	0.0143	0.0152	0.0139	0.0182	0.0151	0.0145	0.0170	0.0163	0.0136	0.0178
$MSE\hat{X}^{(2)}$	86.9164	99.2438	98.0473	102.3034	104.9550	97.5550	95.1184	92.4039	99.9952	98.6594
RMSE	0.8312	0.8404	0.8771	0.8248	0.8244	0.8313	0.8227	0.8250	0.8202	0.8500

4 Conclusions

We have shown that the ICF of the Kalman filter is an alternative algorithm for the Kalman filter that is both mathematically equivalent to the Kalman filter and computationally more efficient for NUC on FPA. This mainly is because the number of observations needed to compensate corrupted block of frames is much greater than the number of system state variables. Further, the inverse covariance filter is better suited for problems where there exists no previous knowledge of the initial system states, fact well suited for practical situations in NUC applications.

It was also shown that the NUC performance of the inverse covariance filter is similar to the Kalman filter. The roughness parameter, the MSE for estimated states variables and the RMSE for the corrected frames are very similar to those obtained using Kalman filter. In offset estimation, the proposed ICF results in low performance than traditional filter, but with minor increments in the length of the block or using sampled frames, the performance of the filter can be improved. The filter, as traditional Kalman filter, is capable to capture the drift in nonuniformity.

The improve of computational efficiency using the ICF of the filter was demonstrated comparing both filters and significative reductions in time are obtained for block lengths greater than 200 frames suiting the algorithm as an excellent option to be calculated on line.

5 Acknowledgments

This work was supported by the 'Fondo Nacional de Ciencia y Tecnología' FONDECYT of the Chilean government. The authors thanks Ernest E. Armstrong at the United States Air Force Research Laboratory, Wright-Patterson Air Force Base and Majeed M. Hayat at University of New Mexico, USA for many valuable comments.

References

- [1] M. Hayat, S. Torres, E. Armstrong, Model Base Real-Time Nonuniformity Correction in Focal Plane Array Detectors, Proc. SPIE, vol. 3377, 1998.
- [2] M. Hayat, S. Torres, E. Armstrong, B. Yasuda, Statistical Algorithm for Non-Uniformity Correction in Focal Plane Arrays, OSA Applied Optics, vol.39, 2000.
- [3] E. Armstrong, M. Hayat, R. Hardie, S. Torres, B. Yasuda, Non-Uniformity Correction for Improved Registration and High Resolution Image Reconstruction in IR Imaginary, Proc. SPIE, vol. 3808, 1999.
- [4] S. Torres, M. Hayat, E. Armstrong, B. Yasuda, On the Performance Analysis of a Recent Statistical Algorithm for Non-Uniformity Correction in Focal Plane Arrays, Proc. CISST, vol. 3703, 2000.
- [5] S. Torres, M. Hayat, E. Armstrong, B. Yasuda, A Kalman-Filtering Approach for Non-Uniformity Correction in Focal-Plane Array Sensors, SPIE vol. 4030, 2000.
- [6] R. Hardie, M. Hayat, E. Armstrong, B. Yasuda, Scene Based Non-Uniformity Correction Using Video Sequences and Registration, OSA Applied Optics, vol. 39, 2000.

- [7] S. Torres, M. Hayat, Compensation for Gain and Bias Nonuniformity and Drift in Array Detectors: A Kalman-Filtering Approach, Submitted to IEEE Trans. Image Processing, Feb. 2001.
- [8] S. Cain, M. Hayat, E. Armstrong, Projection-Based Image Registration in the Presence of Fixed-Pattern Noise, IEEE Trans. on Image Proc., vol. 10, num. 12, 2001.
- [9] B. Ratliff, M. Hayat, R. Hardie, An Algebraic Algorithm for Nonuniformity Correction in Focal-Plane Arrays, Proc. SPIE, vol. 4372, 2001.
- [10] G. Holst, CCD Arrays, Cameras and Displays, SPIE Optical Engineering Press, 1996
- [11] M. Schultz, L. Caldwell, Nonuniformity Correction and Correctability of Infrared Focal Plane Arrays, Proc. SPIE, vol. 2470, 1995.
- [12] G. Minkler, J. Minkler, Theory and Application of the Kalman Filtering, Magellan Book Company, 1993.
- [13] D. Scribner, M. Kruer, J. Killiany, Infrared Focal Plane Array Technology, Proc. of the IEEE, vol. 79, 1991.
- [14] A. Milton, F. Barone, M. Kruer, Influence of Nonuniformity on Infrared Focal Plane Array Performance, SPIE Opt. Eng., vol. 24, 1985.

Adaptive Bias Compensation for Non-Uniformity Correction on Infrared Focal Plane Array Detectors

Esteban Vera R. Rodrigo Reeves D. Sergio Torres I.

Department of Electrical Engineering, Universidad de Concepción, CHILE

Abstract. The non-uniform response in infrared focal plane array (IRFPA) detectors produces corrupted images with a fixed-pattern noise. In this paper we present a new adaptive scene-based non-uniformity correction (NUC) technique. The method simultaneously estimates detector's parameters and performs the non-uniformity compensation using a neural approach and a Kalman estimator in a frame by frame recursive basis. Each detector's output is connected to its own inverse model: a single 1-input linear neuron. The neuron bias is directly related to the detector's offset, and have the property of being softly adapted using simple learning rules, choosing a suitable error measure to fit the NUC objective. The proposed method has been tested with sequences of real infrared data taken with a InSb IRFPA, reaching high correction levels, reducing the fixed pattern noise, and obtaining an effective frame by frame adaptive estimation of each detector's offset.

1 Introduction

Infrared imaging systems are widely used in a variety of real applications like remote sensing, astronomy, surveillance, etc... Most of the infrared imaging sensors are based on the **InfraRed Focal Plane Array (IRFPA)** technology. An **IRFPA** consists of an array of independent infrared detectors aligned at the focal plane of the imaging system [1]. Unfortunately, each detector has unequal responses under the same stimulus, which leads to the presence of a fixed-pattern noise in the images. This **IRFPA** non-uniformity response problem is even more challenging due to the slow and random drift of the fixed-pattern noise.

The best calibration methods for the Non-Uniformity Correction (NUC) of an **IRFPA** are based on the use of uniform infrared sources. These methods are denominated as Reference-Based NUC techniques. The most used and simplest one is the *Two-Point Calibration* method [2, 3], which employs 2 blackbody sources at different temperatures to calculate the gain and offset detector's parameters. Unfortunately, such kind of methods needs to halt the normal operation of the system.

Recently, there have been an increasing research field that postulates that all the disadvantages of the Reference-Based methods can be avoided if the information of the captured scene images is used to estimate the required parameters to perform NUC. These methods are known as Scene-Based NUC techniques. The first scene-based NUC method was developed by Narendra [4] and then it was implemented by Harris et al.[5]. It associates the first and second statistical moments of each pixel with the gain and offset parameters. Hayat et

al.[6] have developed a novel statistical algorithm, which assumes that the input irradiance is a uniformly distributed random variable. This assumption decreases the amount of frames needed for a good statistical inference of the moments. Instead of estimate the detector's parameters, Scribner et al.[7] proposed a least-mean-square error (LMS) adaptive temporal high-pass filtering to remove the fixed pattern noise.

Returning to the parameter estimation techniques, in [8, 9] image registration techniques are used to track pixels shifts between frames, calculating then the associated parameters for the detectors involved in the detected shifts, using an equation system to solve the problem. A novel but resource intensive method is proposed in [10], where a Kalman filter applied to a block of frame data is used to estimates the gain and offset for the detectors. Following with the statistical estimation, Torres et al.[11] have developed a novel adaptation for the constant statistics algorithm applied to uniform distributed infrared data.

All of the mentioned Scene-Based techniques make the assumption that all the detector's parameters remain constant over a certain block of frames, and its performances could decrease depending on the parameters drift. Clearly, none of such techniques adapts sensor's parameters over time under a frame by frame basis. The first approach in this way was first developed in [12], where a retina-like neural net is used to perform the non-uniformity compensation. Following this effort, but in a different way, in this paper a novel algorithm is proposed, where the mean of each pixel is forced to equate the global image mean, then the effect of each detector's offset on the output image is near the needed compensation level.

To perform the proposed adaptive NUC, an array of linear neuron's must be connected to each detector's output, acting as the inverse model for each detector. The parameters of the neurons are smoothly updated using the backpropagation of some error measure. To compute this error, we compare the output of each neuron with the overall input data mean, calculated with a recursive Kalman estimator. In resume, the used error measure is based on the equalization of the mean of all the detectors data at the global mean of the input image. In this way, is expected that this technique reduces drastically the fixed-pattern noise and may also helps to eliminate dead and saturated pixels, which sometimes are the maximum expression of the presence of non-uniformity.

Finally, this paper is organized as follows. In Section 2, a review of the non-uniformity problem on IRFPA's, and the most common correction techniques are presented. Next in Section 3, the Adaptive NUC method here proposed is detailed and explained. In Section 4, the proposed technique is applied to a set of real infrared data and some results and comparisons are shown. Finally, in Section 5, the final remarks and conclusions are given.

2 Non-uniformity and NUC

In order to characterize the non-uniformity problematic, is quite useful to present the single detector's model as the unit cell to be studied. Thus, assuming that an infrared detector is constituted by a photodiode working in its linear zone, for the $(ij)^{\text{th}}$ detector in the focal plane array, the measured signal Y_{ij} at a given time n can be expressed as:

$$Y_{ij}(n) = a_{ij}(n) \cdot X_{ij}(n) + b_{ij}(n) \quad (1)$$

where $a_{ij}(n)$ and $b_{ij}(n)$ are the gain and offset of the ij^{th} detector, and $X_{ij}(n)$ is the real incident infrared radiation collected by the respective detector.

Then, the non-uniformity response problem can be understood as the disparity between the gain and offset of the detectors. So, the major focus of the non-uniformity correction

(NUC) algorithms is to estimate, in some way, these gain and offset parameters from the readout values Y_{ij} . Thus, the real infrared radiation $X_{ij}(n)$ can be found using the following correction equation:

$$X_{ij}(n) = \frac{Y_{ij}(n) - b_{ij}(n)}{a_{ij}(n)} \quad (2)$$

The resultant correction quality relies on an accurate parameter estimation. Moreover, as the parameters drifts slowly and randomly over time, the estimation must also be performed continuously to maintain and assure a good NUC performance.

2.1 Two Point Calibration

The two-point calibration is by far the most used reference-based NUC technique. To perform it, only 2 different uniform radiation sources (blackbody radiators) are needed. After the interruption of the **IRFPA** imaging system to acquire the images correspondent to each blackbody, a 2-equation system is formed for each pixel where the readout values Y and the original radiation X are known. Then the gain a and offset b can be calculated, and the corrections can be performed using equation (2). The estimated parameters using the two-point calibration procedure are the best possible approximations for the real detector's parameters. But unfortunately an estimation for dead or saturate pixels correction are not provided, due to the undetermined equation system produced for these pixels. On the other side, like all the NUC techniques, the calibration must be realized periodically in order to compensate possible parameter drifts, so a constant stop in the normal operation of the imaging system must be made.

2.2 Constant Statistic Constraint

This method, well implemented by Harris et al. in [5], is widely used for its simplicity. It uses the scene image information (readout data) to estimate the detectors parameters. To achieve the estimation, either the incident infrared radiation X as well as the readout infrared radiation Y are considered as gaussian random variables. Then, is assumed that for all the pixels in the **IRFPA** the first and second statistical moments of the real infrared radiation values X are the same. Moreover, the incident radiance is forced to be a zero mean and unitary standard deviation statistical distribution.

Then, the mean and variance must be calculated for each pixel through time. Then, these results are used to estimate the detector's parameters, associating the mean with the offset and the standard deviation with the gain. Unfortunately, in order to achieve stable statistics for a proper parameter estimation, large block of frames must be used. Thus, after the estimation and further correction, where the results for the real radiance X are centered at zero, and must be properly scaled to fit the same range as the uncorrected readout data.

3 Proposed Method

The graphical representation for equations (1) and (2) for a single detector are presented in figure 1 a) and b) respectively. Once the gain and offset parameters are obtained from the readout Y value from the model in figure 1.a), the corrected infrared radiance values X can be calculated using the inverse linear model of figure 1.b) using the same parameters.

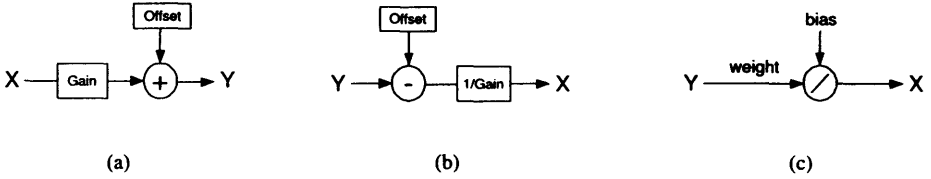


Figure 1: FPA detector's models. a)Linear model; b)Inverse linear model; c)Neural model.

As can be observed, the inverse detector model presented in figure 1.b) can be perfectly represented by the linear neuron presented in figure 1.c), where there is a simple association between the detector's parameters gain and offset with the neuron parameters weight w and bias b . If these neuron parameters are obtained in order to perform the compensation of the non-uniformity, then the detector's parameters are also found.

If the neural model is used, its parameters can be estimated through soft-computing techniques, like backpropagation or the delta rule [13], following the optimization/minimization of a given error measure or functional. Thus, the use of neuron nodes can simultaneously estimate the needed parameters and perform the desired corrections, allowing the implementation of an online self-calibration NUC method in a frame by frame basis.

The use of a linear neuron as the basis of the proposed NUC method has also the following advantages. First, is the one which best matches to the inverse model of a single sensor. Second, the error is directly propagated to update the parameters when backpropagation is used. And finally, it allows the use of the raw data, without any pre or postprocessing.

Focusing on the NUC problem, it's well known that in most cases the offset non-uniformity dominates over the gain one. Then, it was decided to leave the gain as a constant and unitary value to simplify the estimation process. Thus, our NUC problem is reduced to estimate only each neuron's bias. To make those estimations useful to perform NUC, the bias must be updated in the sense where the difference between the neuron's output value and the input global mean is minimized. This will force all the output values to tend in an average way to the same global mean, and thus the fixed pattern noise will be reduced. In other words, an adaptive mean equalization will be performed, instead of the static equalization of the mean stated in [5]. The main hypothesis in the proposed method is that the mean value of the read-out pixels is a measure of the possible offset present.

A scheme of the complete proposed method for (NUC) is presented in the figure 2. As can be seen, the read-out values of the uncorrected image pixels are the inputs for the array of linear nodes. Then, after being summed with the correspondent bias level, a compensated pixel value is reflected at the output of each node (due to the linear activation function of the node). On the other side, a Kalman filter is estimating in a parallel way the temporal mean values for each pixels from the read-out values [14]. Then, after the average process off all the pixels mean, a Global Mean Estimation of the image is obtained. This last value becomes the reference (or desired output) to be compared with all the corrected pixel values at the output of each node. The error e produced in (3) is then backpropagated, and the bias b value for each node is updated with a proportional value, as it's expressed in (4). The supposed weight value that appears in figure 1.c) is omitted in the final scheme as it is kept as a constant

equal to 1, because only offset estimation/compensation is required. The steepest descent-like algorithm to be used for the bias recursive update is thus expressed as follows:

$$e_{ij}(n) = d(n) - X_{ij}(n) \quad (3)$$

$$b_{ij}(n+1) = b_{ij}(n) + \eta e_{ij}(n) X_{ij}(n) \quad (4)$$

where $d(n)$ is the desired output and $X_{ij}(n)$ is the ij^{th} node output's. The learning rate η is responsible to control the update velocity for the bias. All the calculations are performed separately for each node, and there's only one common value to all the neurons: the averaged estimated input global mean.

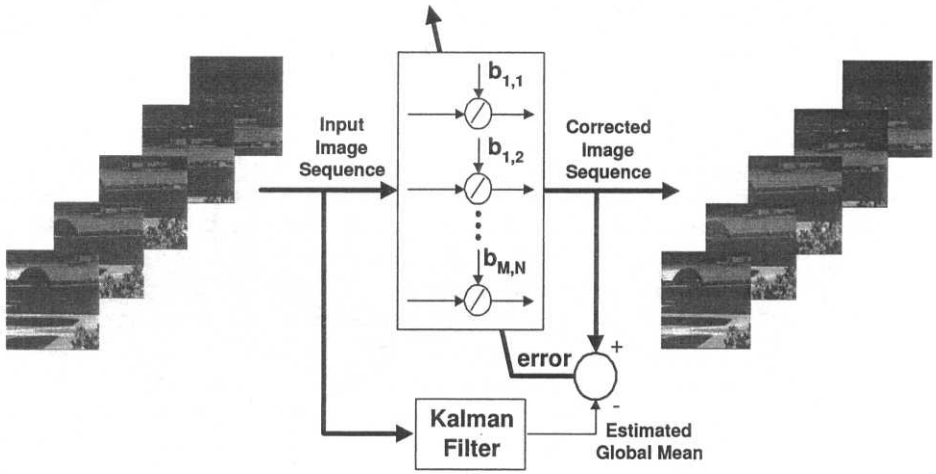


Figure 2: Soft-Adaptive NUC Scheme.

4 Applications to Real Infrared Data

The proposed NUC scheme is implemented in Matlab language and is tested on five real infrared image sequences captured at five different times in one day. The data sets have been taken using a 128x128 InSb FPA camera (Amber Model AE-4128) operating in the 3-5 μm range and at a rate of 30 frames per second. For the first data set (taken at 1:00 PM), a two-point calibrated version of the image sequence is also implemented.

4.1 Non-Uniformity Correction Performance Measurements

Image quality measurements on the performed corrections are crucial to determine the effectiveness of a given NUC method. In this paper, the **Root Mean Squared Error (RMSE)** and the **Universal Image Quality Index** are used to measure the NUC capability of the proposed

method. The **RMSE**, in an average sense, is the Euclidean mean distance in a pixel by pixel basis between two images x and y as follows:

$$RMSE = \frac{1}{n} \sqrt{\sum_{ij}^n (x_{ij} - y_{ij})^2} \quad (5)$$

where in this case x can be the corrected or uncorrected images, y is the reference two point calibrated image, and n is the number of pixels in the image frame. Unfortunately, this measure isn't so good to evaluate the quality of a given correction in a perceptual way. Sometimes all the corrected images could be biased at the same amount, and the RMSE will be high, but in the reality the quality of the corrected image is very good and doesn't present the unwanted fixed pattern noise. For this reason, we also use of the *Universal Image Quality Index (Q)* [15] to evaluate the relative quality of the corrected images. This index compares two images x and y in the structural distortion sense, and it is expressed in the following equation:

$$Q = \frac{\sigma_{xy}}{\sigma_x \sigma_y} \cdot \frac{2\bar{x}\bar{y}}{\bar{x}^2 + \bar{y}^2} \cdot \frac{2\sigma_x \sigma_y}{\sigma_x^2 + \sigma_y^2} \quad (6)$$

where \bar{x} , \bar{y} , σ_x and σ_y are the mean and standard deviation of both images, and σ_{xy} is the correlation between the images. Thus just as stated, this quality index models any distortion as a combination of three different factors: loss of correlation, mean distortion and variance distortion.

Four thousand frames of the 1 PM sequence were corrected using the proposed adaptive algorithm using different values for the learning rate η . As no comparable values between the corrected and the uncorrected sequences were obtained by the **RMSE** index, the **Q** index were calculated for all the sequences, using the two-point calibrated sequence as the reference. The results for some of the most representative learning rate results are shown in figure 3. The quality index for the uncorrected image sequence is also displayed. An index value near one indicates that this image, or frame in particular, is very similar to the calibrated reference frame.

As can be seen in the graphic, the Q calculated for large learning rates on the corrected data sequence performs better than the uncorrected one only in some frame intervals, and even competes with the other corrected sequences. For median learning rates, the performance is very good only in the first half of the infrared sequence. For small learning rates, the quality index is always better than the uncorrected one.

To certify the behavior of the quality index, several video sequences were generated from the corrected sequences obtained with different learning rates, and then visually compared with the two-point calibrated version and the correspondent uncorrected sequence. It is seen from the videos for large learning rates that the quality peaks coincide directly with quick motion of the camera, and very good corrections are produced. But when the camera stays in the same position, the corrected images tend to disappear. The same, but in a soft way, happened to the median learning rates. For small learning rates, the corrected video sequence takes a long time to eliminate the fixed pattern noise, but always produce softer images than the uncorrected sequence, and under soft movements the corrections are very close to the original calibrated ones. When no movement is produced, ghosting artifacts starts to appear in the following moving frames, but the effect is smoothly compensated over time.

A graphic which shows the relative average for **Q** and **RMSE** over the whole sequence of data is presented in figure 4. This graph represents how many times a corrected sequence

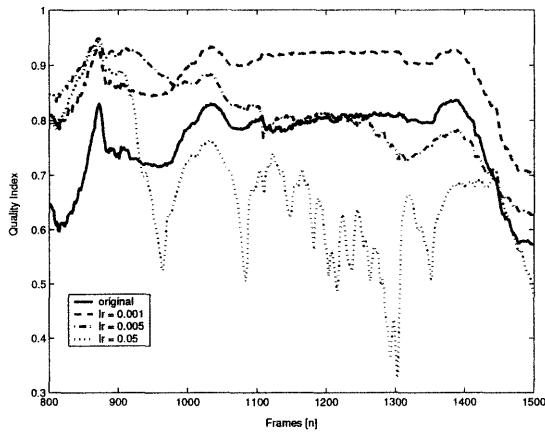


Figure 3: Quality index Q for the original raw frames and the corrected versions at various learning rates.

has improved with respect to the uncorrected sequence. It can be noticed that better quality is achieved with a learning rate of 0.0025, but as the error measure starts to increase too much, the learning rate value of 0.001 is selected as the optimum one to perform the corrections.

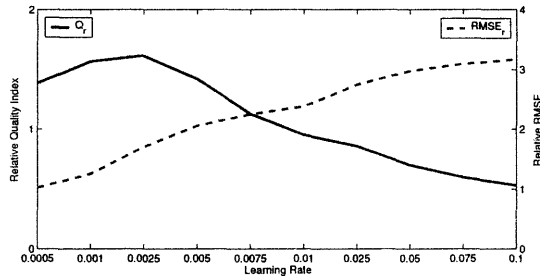


Figure 4: Mean value for the relative quality index Q_R and the relative $RMSE_r$ v/s different learning rates.

Using the selected optimal learning rate stated above, a comparison between its calculated quality index and the one obtained with the constant statistics correction technique is performed for the whole data sequence, and is presented in figure 5. The constant statistics results were obtained with the same 4000 frames of data, but calculated in a batch way. To allow this comparison, the constant statistics corrected data must be properly scaled to fit the average and variance of the original data. From the graphic is easy to note that the proposed adaptive method clearly starts with a poor quality as the original data, but then the quality increases together with the frame number, achieving in some parts as high quality results as the ones obtained with the constant statistics NUC method.

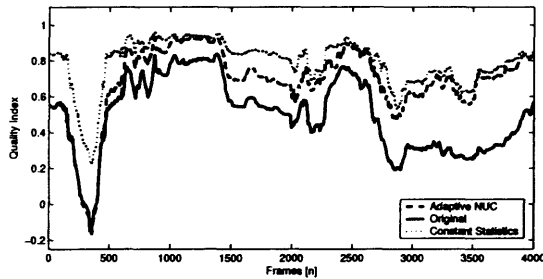


Figure 5: Quality index Q for the original data, adaptive NUC and constant statistics NUC corrected data.

4.2 Non-Uniformity Correction Visual Evaluation

No two-point calibrated images are available to realize performance comparisons at 6:30 AM, 8:00 AM, 9:30 AM and 11:00 AM. Then, these sets have been corrected using the selected optimum learning rate found in the previous subsection. The resulting video sequences were analyzed, and some frames of the uncorrected and corrected sequences are shown in figure 6.

From the video sequences it's concluded that very good results were obtained with the selected learning rates, although similar artifacts, like the ghosting presented in the 1:00 PM data set, were presented when the sequence got too static. The foregoing problem increased with the day time. The best performance for the NUC method was visualized at earlier data sequences. The good quality of the corrections achieved can be seen in the pictures shown in figure 6.b) and figure 6.d). For the late morning data sets like the 11:00 AM sequence, the overall NUC quality is good, but some ghosting artifacts are visible, as can be perceived in the figure 6.f).

5 Conclusions

An adaptive scene-based NUC method to reduce the fixed pattern noise in infrared image sequences has been developed. The method is based on a scheme that estimates the detector's parameters and perform the correction in a frame by frame basis without any previous assumptions or knowledge of the raw data. In addition, due to its recursive and adaptive nature, an efficient online implementation can be easily retrieved. Moreover, the presented scheme doesn't need the help of any additional processing or data scaling as other algorithms.

To measure the efficiency of the proposed NUC technique, a novel quality index has been computed on the corrected frames. This metric has been applied to the 1 PM tested sequence, and quality comparisons have been made with correction generated using different learning rates. At the optimal selected learning rate, the quality index shows to be always better than the uncorrected sequence, and sometimes as good as the obtained with the popular constant statistics technique. But sometimes there are some unwanted ghosting artifacts in the images, related to the presence of large static scenes, which obviously decreases the quality measures. Better results, at least visually, were achieved with the earlier hours data sets using the same optimal learning rate. This effect is mainly produced because the images at such hours are dominated by the fixed pattern noise instead of the infrared radiation values.

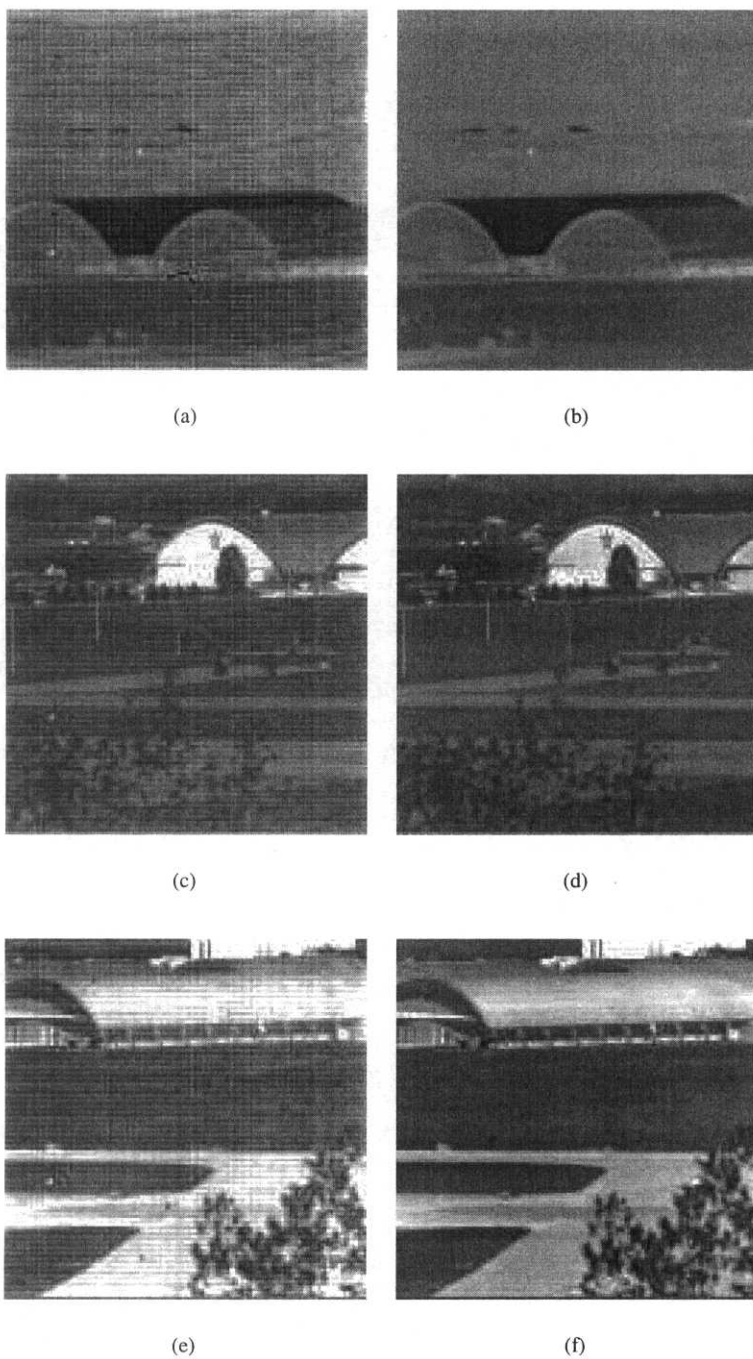


Figure 6: Corrected image samples with the proposed algorithm. a) and b) show respectively the raw frame and its correction at 6:30AM. c) and d) at 9:30AM. e) and f) at 11:00AM.

Further research must be done to improve the algorithm performance, manipulating adaptively the learning rate to reduce the ghosting and making use of some possible spatial correlation to speed up the estimations.

6 Acknowledgments

This work was supported by the “Fondo nacional de ciencia y tecnologia FONDECYT” of the Chilean government. The authors thanks Ernest E. Armstrong at the US Air Force Research Laboratory in the Wright-Patterson Air Force Base, for assistance in collecting the Infrared data. The authors also thanks Dr. Majeed M. Hayat at the Electrical and Computer Engineering Department of the University of New Mexico, USA, for many valuable comments.

References

- [1] D. Scribner, M. Kruer, and J. Killiany. Infrared focal plane array technology. *Proceedings of the IEEE*, 79(1):66–85, 1991.
- [2] D. Scribner, M. Kruer, and C. Gridley. Physical limitations to nonuniformity correction in focal plane arrays. *Proceedings of SPIE*, 865:185–201, 1987.
- [3] J. Mooney, F. Shepherd, W. Ewing, J. Murquia, and J. Silverman. Responsivity nonuniformity limited performance of infrared staring cameras. *Optical Engineering*, 28:1151–1161, 1989.
- [4] P. Narendra. Reference-free nonuniformity compensation for ir imaging arrays. *Proceedings of SPIE*, 252:10–17, 1980.
- [5] J. Harris and Y. Chiang. nonuniformity correction of infrared image sequences using the constant-statistics constraint. *IEEE Transactions on Image Processing*, 8(8):1148–1151, August 1999.
- [6] M. Hayat, S. Torres, E. Armstrong, S. Cain, and B. Yasuda. Statistical algorithm for nonuniformity correction in focal-plane arrays. *Applied Optics-IP*, 38(5):772–780, February 1999.
- [7] D. Scribner, K. Sarkady, J. Caulfield, M. Kruer, G. Katz, and C. Gridley. Nonuniformity correction for staring focal plane arrays using scene-based techniques. *Proceedings of SPIE*, 1308:224–233, 1990.
- [8] R. Hardie, M. Hayat, E. Armstrong, and B. Yasuda. Scene-based nonuniformity correction with video sequences and registration. *Applied Optics-IP*, 39(8):1241–1250, March 2000.
- [9] S. Cain, M. Hayat, and E. Armstrong. Projection-based image registration in the presence of fixed-pattern noise. *IEEE Transactions on Image Processing*, 10(12):1860–1872, December 2001.
- [10] S. Torres and M. Hayat. Compensation for gain and bias nonuniformity and drift in array detectors : A kalman-filtering approach. *IEEE Transactions on Image Processing*. under review.
- [11] S. Torres, R. Reeves, and M. Hayat. Scene-based non-uniformity correction method using constant range: Performance and analysis. *Proceedings of the 6th SCI*, IX:224–229, July 2002.
- [12] D. Scribner, K. Sarkady, M. Kruer, J. Caulfield, J. Hunt, M. Colbert, and M. Descour. Adaptive retina-like preprocessing for imaging detector arrays. *Proceedings of the IEEE International Conference on Neural Networks*, 3:1955–1960, 1993.
- [13] J. Principe, N. Euliano, and W. C. Lefebvre. *Neural and Adaptive Systems*. John Wiley & Sons, 2000.
- [14] E. Vera. Optimal and recursive mean estimation for infrared focal plane array sensors. Technical report, Departament of Electrical Engineering, Universidad de Concepción, 2002.
- [15] Z. Wang and A. Bovik. A universal image quality index. *IEEE Signal Processing Letters*, 9(3):81–84, March 2002.

Combining Genetic Algorithms and Neural Networks to Build a Signal Pattern Classifier

Roshdy S. Youssif and Carla N. Purdy

ECECS Dept., University of Cincinnati, Cincinnati, Ohio 45221-0030, USA
email: youssir@email.uc.edu, carla.purdy@uc.edu

Abstract. In this paper we show how genetic algorithms and neural networks are combined to build a high performance Signal Pattern Classifier (GNSPC). Signal patterns are intrinsic to many sensor-based systems. The goal of GNSPC is to differentiate among large numbers of signal pattern classes with low classification cost and high classification performance. Classification performance is measured by the correct classification of noisy signal patterns despite using pure signal patterns for building the classifier. GNSPC is basically a decision tree classifier with similarity classification rules. The rules are used to test the similarity of signal patterns. A combination of a genetic algorithm and a neural network is used to find the best rules for the decision tree. This combination provides powerful classification capabilities with great tuning flexibility for either performance or cost-efficiency. Learning techniques are employed to set the genetic algorithm global parameters and to obtain training data for the neural network.

1. Introduction

Pattern classification is one of the active areas of research with many developed techniques. Various neuro-fuzzy methods [1] have been developed to classify and cluster different types of patterns in many applications. Genetic algorithms are also used to evolve classifier systems [3] that are useful in different applications. Most of these techniques utilize training data containing noisy versions of the targeted patterns to build the classifier system in order for it to be able to correctly classify noisy patterns. However, in many practical applications the available pattern data for training the classifier system is collected under lab conditions with no or negligible amounts of noise. Contrarily, the classifier is expected to perform adequately under field conditions with variable noise levels.

In this paper we present a Genetic Neural Net Signal Pattern Classifier GNSPC that is built with pure pattern data and tested against noisy data. The goal of GNSPC is to be able to differentiate among large numbers of classes (for example hundreds of classes), with high performance and efficient classification cost. The main idea of GNSPC is to simplify the classification decision-making process by basing it on similarity measures to reference patterns (rule patterns). Obviously, picking such rule patterns is not an easy task and impacts the performance of the classifier directly. GNSPC combines genetic algorithms and neural networks for finding the rule patterns and the similarity measures. It then utilizes a decision tree for making the classification decision. For the considered application, this combination has demonstrated superior power in achieving the goal over using a single technique such as neural networks, which need to be trained with noisy data to be able to perform well under noisy conditions. An ultimate goal for GNSPC is to use it

as an example of building hybrid intelligent systems and to study ways of coordinating the synergies of these techniques.

Sensor based systems are usually built from a collection of sensors, each dedicated to measuring a certain attribute of the environment or a specific component of the compound to be detected. For example, many electronic nose systems use arrays of gas sensors [7], each sensor with its specific sensitivity to a certain range of gases. Electronic nose systems identify odors by interpreting the patterns of all their sensors' readings as one pattern [7]. The zNose [9] is an example of an electronic nose that uses gas chromatography to identify the component gases of odors and to generate a unified graph for all the components' measurements called the Vapor Print, as shown in Figure 1. Each odor is identifiable by its unique sensors' readings pattern depicted in its Vapor Print.

We call the patterns exhibited in such systems signal patterns. A signal pattern SP can be described as $y = f(t)$ where t is time and y is the sensor reading, e.g., resistance, frequency. Typically, sensor readings are collected every short time step, e.g., milliseconds, over a sampling period ranging from a few seconds to many minutes. The large number of collected readings permits us to approximate y by a continuous function. The full shape of y over the sampling time uniquely identifies the sensed object such as a certain odor.

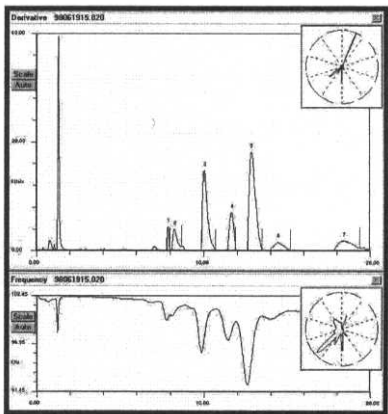


Figure 1. An example of a signal pattern for a perfume produced by zNose [13]. The bottom plot is the signal reading and the top is the derivative of the signal.

Classical pattern classification techniques like ID3 and C4.5 [8] have provided solutions for classifying patterns of a number of discrete or continuous variables. However, they are not directly applicable for classifying large sets of signal patterns (SP sets). Neural networks (NNs) have been applied to similar classification problems but with a limited number of identifiable classes, e.g., 6 perfume classes in [6] and 10 random odor classes in [11]. Decision trees have been combined before with genetic algorithms (GAs) to find the best subset of features [2]. Genetic algorithms were employed to find fuzzy rules in [4] and [10] but not in conjunction with decision trees. In the following sections we discuss the GNSPC structure and how it combines GA and NN to achieve its high performance goal.

2. Structure of GNSPC

GNSPC is basically a binary decision tree. Each node in the tree has a similarity rule SR while each leaf has a class C associated to it. Before building the tree, each SP in

the training set is preprocessed to extract its important features. Starting with a root node, the algorithm recursively adds nodes to the tree while dividing the SP set to pass and fail subsets as in Figure 2. The algorithm finds a similarity rule that best subdivides the SP set associated with the node. A Pass-Child node is added to its parent node and the subset of patterns that satisfy the similarity rule are associated to it. Similarly, a Fail-Child node is added to its parent node.

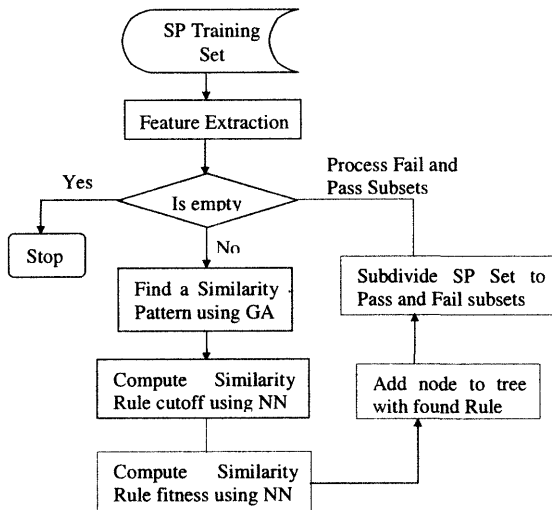


Figure 2. Flow chart of building the classification tree.

After building the tree, classifying an unknown pattern is done by applying the similarity rule at the root to the pattern. If it passes then the Pass-Child node is visited; otherwise the Fail-Child node is visited. This process is repeated till a leaf is reached. A sample classification tree is shown in Figure 3.

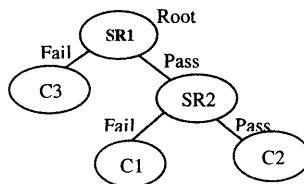


Figure 3. A sample classification tree. Each node has a similarity rule SR. Each leaf has a class C.

2.1. Feature Extraction

To efficiently classify an SP, its important features are extracted and used for classification instead of all its data points. It is desirable to extract the minimum number of features that uniquely identifies the SP and eliminates noise effects. Features can be extracted from the SP directly or from its derivative (first order or higher) if it better accentuates the features (see Figure 1). In GNSPC, feature points are found first and then features are constructed from them.

- *Feature Points* include local maxima and other locations where the SP derivative exhibits a significant change. They can be found by using finite differencing on the SP or by applying a wavelet transform. Points with wavelet coefficients higher than a threshold are taken as feature points. Note that the threshold values for both methods determine the level of detail of the constructed features
- *Features* are constructed from feature points. They can be thought of as the spikes of the signal.

In GNSPC we use features that form unimodal spikes. Each feature is described by its start and end locations, amplitude and area. Other attributes of interest can also be used.

2.2. The Pattern Similarity Rule

In order to specify classification rules, both feature and pattern similarity measures must be developed. For GNSPC we define the feature similarity measure, *Sf*, of 2 features *f1* and *f2* as:

$$Sf = \frac{A(f1) \cap A(f2)}{A(f1) + A(f2) - A(f1) \cap A(f2)},$$

where *A(f1)* is the area of *f1* and *A(f2)* is the area of *f2* and *A(f1) ∩ A(f2)* is defined to be the overlap area between *f1* and *f2*.

The pattern similarity measure *Sp_{1,2}* of 2 patterns *p₁*, *p₂* is the summation of the overlap areas of their features and is defined as

$$Sp_{1,2} = \frac{\sum A(f1) \cap A(f2)}{\sum (A(f1) + A(f2)) - \sum (A(f1) \cap A(f2))}$$

Similarity rules used in the classification tree are defined as:

IF (*S_{r, p}* > *t*) THEN visit the Pass-Child node ELSE visit Fail-Child node,
where *S_{r, p}* is the similarity between the rule's pattern *r* and the unknown SP *p* and *t* is the similarity cutoff. Table 1 shows an example of a similarity rule with its pattern features and similarity cutoff value.

Table 1. An example of similarity rule with a pattern made of 5 features, F1 to F5.

		Start	End	Area	Height	Similarity cutoff
Rule1	F1	43	52	1632	323.03	0.4563
	F2	16	24	1281.1	315.39	
	F3	55	61	1091.1	293.35	
	F4	6	12	970.89	313.262	
	F5	89	95	701.53	218.03	

3. Finding the Similarity Rule

Obviously, the performance of the classifier is very sensitive to the quality of its similarity rules. As mentioned above, the antecedent part of the similarity rule has two main components, the rule's matching pattern and the similarity cutoff value. These two components can be picked arbitrarily with the goal of optimizing the performance of the classifier. The rule's pattern can be constructed from any subset of features from the pattern set associated with the node or any other arbitrary set of features. The rule's similarity cutoff determines the path in the classification tree a certain pattern will take.

In our previous research [12] we used a genetic algorithm to find both the rule's pattern and the similarity cutoff. However, our experiments showed that similarity cutoff

values often fell out of bounds of the similarity values the rule's pattern imposes on the SP set associated with it. This situation resulted in not being able to find rules at some nodes that can differentiate between the patterns of the associated SP set. Hence, on such nodes it was suggested in [12] to delegate pattern classification to another mechanism such as neural networks. But careful analysis of the performance of the previous classifier with noisy data showed that improved performance is achievable if more attention is given to picking the similarity cutoff and computing the similarity rule fitness.

To achieve the high performance goal of GNSPC we have combined a genetic algorithm with a neural network to find the best similarity rule for a given SP set. Basically, the genetic algorithm generates the rule's feature pattern and evolves the best rules while the neural network finds the similarity cutoff for the rule and evaluates the rule fitness. The approach ensures that any generated rule will be able to divide the associated SP set into two subsets, in contrast to [12]. In the following sections we discuss both the genetic algorithm and the neural network employed to find similarity rules in more detail.

3.1. The Genetic Algorithm

As mentioned above, a genetic algorithm [3] is utilized to find the best similarity rule for each node in the classification tree. The rule is made of a pattern r of n features combined with a similarity cutoff value t . The number of features n in the rule can vary among different nodes in the same tree. In the following we present the basic constructs used for the genetic algorithm in GNSPC.

- *Encoding* in GNSPC is done by using strings of real-valued numbers. Each feature in the rule is represented by its attributes, i.e., start and end locations, amplitude and area. The rule string is composed of a sequence of encoded features and a similarity cutoff value at the end. For example, a rule with 5 features will be encoded to a string of 21 real valued numbers.
- *Initial population* in GNSPC is generated by picking m signal patterns from the pattern set associated with the node. Then from each pattern the n features with biggest areas are encoded to make an individual rule string. The rule similarity cutoff t is computed using the algorithm described in the following section.
- *Fitness* is very crucial to the performance of GNSPC. There are many possibilities to measure the fitness of a similarity rule. One fitness measure based on the infomax concept used in decision trees [8] can be stated as: "how evenly does the rule divide the SP set associated with the rule node?" The more evenly the rule divides the pattern set, the higher its fitness score. This will lead to a shallower classification tree and lower classification cost. We have found that simple fitness measures, like infomax, do not yield the desirable high classification performance. Therefore we devised a fitness measure using neural networks as described below.
- *Selection* methods include selection of fittest, roulette wheel selection, and ranking selection. Any of these methods is applicable to GNSPC; however our current implementation uses roulette wheel.
- *Crossover* methods that apply to real numbers can be used in GNSPC since the classification rules are encoded as real number strings. Note that the Lamarckian concept [3] of controlled children is also applicable in GNSPC, especially with crossover methods that produce rules with repeated or greatly overlapping features.
- *Mutation* in GNSPC is done by using non-uniform mutation; however any of the mutation methods applicable to real number strings can be used as well.
- *Termination* condition in GNSPC can be set to a certain level of rule fitness or a limit on the number of generations. Note that the termination condition can be different for different nodes in the tree. For example, higher nodes in the tree may use higher fitness

to achieve better classification performance while lower nodes may use a fixed number of generations to achieve efficiency of computation.

3.2. Finding the Similarity Cutoff

To design an algorithm that finds the best similarity cutoff for a given rule pattern, one needs to study the role of the similarity cutoff in the overall classifier performance. Basically, each rule's feature pattern determines a similarity distribution on the set of patterns it is supposed to differentiate. This similarity distribution can also be viewed as a set of clusters of the similarity values. The similarity cutoff can be chosen at any point along the similarity distribution line $[0,1]$. However, good similarity cutoffs would rather lie on gaps between clusters of the similarity values. Since our classifier uses binary trees, i.e., every rule divides the associated SP set into pass and fail subsets, we look at similarity values as forming two clusters on the sides of the chosen similarity cutoff as in Figure 4. The best similarity cutoff is the one that lies between two clusters with the highest quality. Clusters' quality can be measured by the width of the gap between the two clusters, their relative size and the spread of the two clusters.

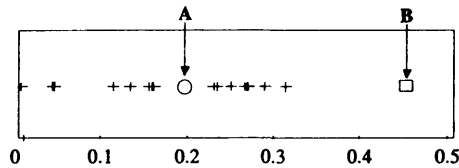


Figure 4. An example of similarity distribution. A is the cutoff picked by the criteria (forms two clusters) and B is a randomly picked cutoff (forms one cluster only).

A good set of criteria for selecting the similarity cutoff for GNSPC is to choose it such that it forms two clusters of almost equal size, with minimum clusters' spread and widest possible cluster gap; also it should almost lie in the middle of the gap. Such criteria would ensure better classification performance on noisy patterns since noise would tend to perturb the pattern similarity measure. Note that the same criteria can be used to evaluate the quality (or fitness) of the similarity rule. The rule fitness can be stated as: "what is the quality of the two clusters formed by the similarity cutoff and how well does the cutoff lie between the two clusters?" It is apparent that evaluation of such criteria is more involved. In our system we are using neural networks for such evaluation as discussed in the following section.

Our algorithm for picking the similarity cutoff and computing the rule fitness is as follows:

- For all SP in the rule set
 - Compute the similarity value
- Sort the gaps between similarity values by the widest
- For the widest k (a chosen value, e.g. 3) gaps
 - Compute the fitness of the middle point of the gap
- Pick the middle point with the highest fitness value as the rule similarity cutoff

3.3. Fitness Evaluation Using Neural Networks

Fitness of the similarity rule is very crucial to evolving good rules. The performance of the classifier under noisy conditions is very sensitive to the quality of the similarity rules selected. As mentioned above, many factors that contribute to the quality of

the rule are related to the quality of the two clusters of pattern similarities the rule imposes on its associated SP set. Factors considered for GNSPC are: the width of the two clusters gap, the relative size of the two clusters, and the clusters' spreads. Note that these factors are functions of the size of the SP set associated with the rule. For example, when the SP size is small, the relative cluster size factor should be less than the cluster gap width factor. On the other hand, when the SP set size is big, the clusters' spread factor should be less than the cluster relative size factor so that bushier classification trees are favored. In general, it is not easy to formulate a fitness function that takes into account all such factors.

We propose using the following formula for computing the similarity rule fitness f :

$$f = c_1 * g + c_2 * r - c_3 * (s_1 + s_2) \quad ,$$

where g is the size of the gap between the two clusters on the sides of the rule similarity cutoff, r is the ratio of the smallest cluster size to the size of the SP set, s_1 and s_2 are the spread widths of the two clusters. Obviously, the difficult part is determining the values of the constants c_1 , c_2 and c_3 .

Neural networks are best suited to approximate complex functions and to improve their approximation with learning from examples. We chose a generalized regression network GRNN, a type of radial basis network, for the approximation of the function relating constants; c_1 , c_2 and c_3 to SP set size. Using sample SP sets of different sizes along with their noisy versions, constants data can be generated to train the GRNN. For each SP set a number of classifiers would be built each using different constant values for the fitness function. The constants that produce the classifier with the highest performance on the relevant noisy SP sets would be used as target training values for the GRNN for that SP set size. We rely on the GRNN's function approximation ability to estimate the constants values for any SP set size.

3.4. Setting the Classifier Parameters

Building the classification tree can be done in batch mode where the training set is supplied to the algorithm at once or in incremental mode where the training patterns are supplied in subsets. In both ways learning techniques can be used to find the best values for GNSPC parameters as depicted in Figure 5. Some GNSPC parameters are the number of features per classification rule, the number of generations to search for the rule, and the fitness function constants.

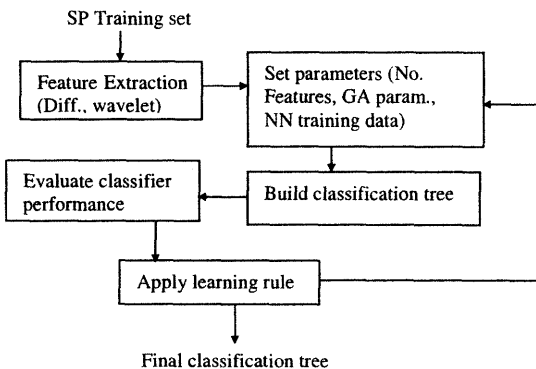


Figure 5. Applying the learning rule to the process of building the classification tree.

4. Experiments and Results

GNSPC has been implemented using MATLAB due its ease of use, available neural network toolbox and genetic algorithms function approximation implementation GAOT [5]. A randomly generated SP set that mimics Figure 1 has been used to enable us to test the algorithm on a large set of patterns. SP parameters such as maximum amplitude, duration, and maximum number of spikes are set to ranges similar to zNose data. Feature points are detected by using finite differencing while features are constructed from sequences of feature points forming unimodal spikes as in Figure 6. The genetic algorithm used real valued number encoding with geometric crossover and non-uniform mutation.

A preliminary testing for GNSPC was conducted on a set of 48 random signal patterns. Note that previous work has used smaller number of classes, e.g. 6 perfume types in [6], 10 random odors in [11] and 24 random signal patterns in [12]. The results of this preliminary test demonstrate the ability of GNSPC to correctly differentiate among the 48 classes with minimal computation evident by small population size (8), few generations (8) and small number of features per rule (5). GNSPC was able to correctly classify each pattern in the 48-SP set to itself.

4.1. Classifier Performance

To test the performance of GNSPC, we generated 12 sets of perturbed versions of each pattern in the 48-SP set. An example of a pattern and one of its perturbed versions is shown in Figure 6. We used amplitude perturbations aiming to mimic noise typically experienced under field conditions. The performance was measured as the average percentage of correctly classified patterns in the perturbed sets.

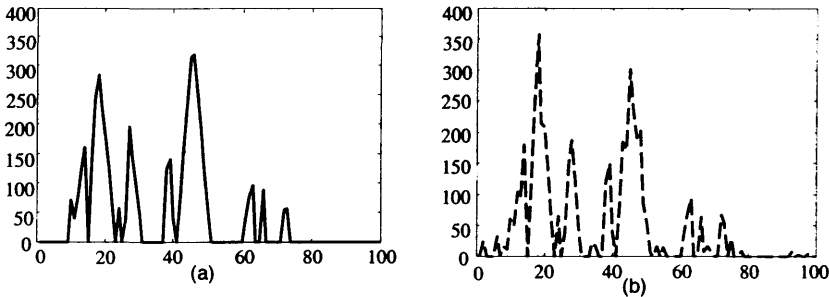


Figure 6. Example of a signal pattern (a) and its perturbed version (b).

Prior to testing the performance of GNSPC we ran tests on SP sets of different sizes, ranging from 4 to 40 with different values of the cluster size and cluster spread constants of the fitness function mentioned above. The performance of each classifier was tested using 5 perturbed sets of the original SP set used to build the classifier. The constants values that produced the highest performance classifier were used as the target values for training the GRNN for that SP set size. Figure 7 shows a plot of the cluster size and cluster spread constants relative to the SP set size as simulated by the GRNN after training. As expected when SP set size is small such constants have lower values and when SP set size is large the clusters' size constant has higher values than the clusters' spread constant.

In order to evaluate the effect of using GRNN on the performance of GNSPC we tested another classifier built by the same procedure as GNSPC but without using the GRNN. Instead of computing the rule similarity cutoff by the GRNN, the cutoff was obtained by the genetic algorithm and the fitness was measured by the relative size of the

Pass and Fail subsets produced by the similarity rule as in [12]. When the Pass and Fail subsets sizes are equal the rule has highest fitness. Figure 8 shows the performance of GNSPC on the 12 perturbed 48-SP sets in solid bars and the performance of the other classifier [12] in dashed bars. Our results show that GNSPC is superior in performance with noisy data as expected due to picking better similarity cutoff values for similarity rules and evaluating the rule fitness more precisely using GRNN. We are currently fine tuning GNSPC parameters to improve its overall performance.

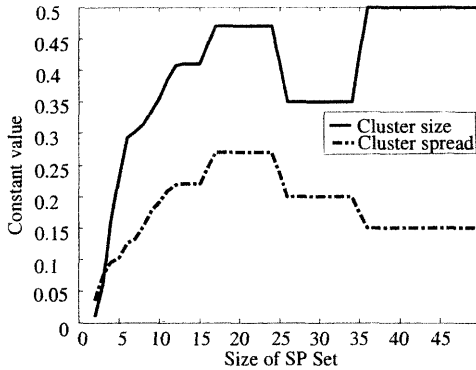


Figure 7. Cluster size and cluster spread factors as simulated by the GRNN.

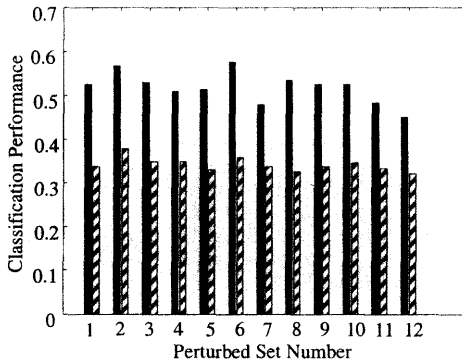


Figure 8. Classification performance of GNSPC on 12 perturbed sets of 48 signal patterns (solid bars) compared another classifier not using GRNN (dashed bars).

The cost of building GNSPC is mainly dependent on the number of the nodes in the final classification tree. The cost of the preliminary step of feature extraction is insignificant compared to the cost of building the classifier. Finding the similarity rule for each node involves running the genetic algorithm to get the best similarity pattern and the neural network to determine the similarity cutoff. Limiting the number of generations for evolving the similarity pattern controls the cost of the genetic algorithm. The cost of determining the similarity cutoff involves the cost of evaluating the similarity of the SP set associated with the rule node and the candidate rule patterns. However, pattern similarity evaluation involves simple number comparisons.

Basically, all the costly operations are needed only for building the GNSPC. On the other hand, the cost of classifying a pattern is very cheap. It is composed of extracting the

features of the pattern then, finding the leaf in the tree where the pattern is matched. Finding the class leaf is mainly dependent on the depth of the classification tree and the number of features in the pattern. The shallower the tree the fewer comparisons are needed to reach the leaf. That is the reason for favoring similarity rules that divide the related pattern set evenly. Pattern similarity evaluation is a cheap operation as mentioned above.

5. Conclusions and Future Work

We have shown in this paper a way of combining different intelligent techniques to build a powerful signal pattern classifier. We believe that combining more than one technique is necessary to solve complex problems. Each technique has its effectiveness in solving part of the problem. Genetic algorithms are useful in searching the large space of similarity rules and neural networks in computing the rule similarity cutoff and evaluating its fitness. Learning techniques aid in adaptively setting the system overall parameters and gauging the convergence of the classifier performance. We are currently refining the system to improve results on larger sets of SP classes with different types of simulated noise conditions.

We are currently working on using fuzzy logic instead of the crisp similarity rules for improving the classification performance. In the future, we would like to experiment with wavelet transforms for better feature extraction with different levels of details. Study of the effect of Lamarckian evolution on the classifier performance will also be undertaken. There are many parameters for the genetic algorithms that need to be investigated. Finally, we would like to apply the classifier to other classes of problems such as image object tracking.

6. References

- [1] S. Abe, *Pattern Classification: Neuro-fuzzy Methods and Their Comparison*, Springer-Verlag, 2001.
- [2] J. Bala, J. Huang, H. Vafaie, K. DeJong and H. Wechsler, "Hybrid Learning Using Genetic Algorithms and Decision Trees for Pattern Classification", *IJCAI*, Montreal, 1995.
- [3] Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
- [4] F. Herrera, M. Lozano, and J. L. Verdegay, "Generating fuzzy rules from examples using genetic algorithm", In B. Bouchon-Meunier, R. R. Yager, and L. A. Zadeh, editors, *Fuzzy Logic and Soft Computing*, pages 11--20. World Scientific, 1995.
- [5] C. Houck, J. Joines and M. Kay, "A Genetic Algorithm for Function Optimization: A Matlab Implementation", Technical Report NCSU-IE TR 95-09, North Carolina State University, 1995.
- [6] B.G. Kermani, S. S. Schiffman, and T. Nagle, "Using Neural Networks and Genetic Algorithms to Enhance Performance in an Electronic Nose", *IEEE Trans. on Biomedical Engineering*, vol. 4, no. 4, pp. 429-439, 1999.
- [7] H. T. Nagle, S. Schiffman, and R. Gutierrez-Osuna, "The How and Why of Electronic Noses", *IEEE Spectrum*, pp. 22-33, Sept. 1998.
- [8] J. Quinlan, "Induction of Decision Trees", *Machine Learning*, vol. 1, pp. 81-106, 1986.
- [9] E. Staples, "Virtual Chemical Sensor Arrays and High-Resolution Olfactory Images - The zNose", *Sensors Magazine*, www.sensormag.com, vol. 18, no. 6, June 2001.
- [10] W. M. Spears and K. A. De Jong, "Using Genetic Algorithms For Supervised Concept Learning", *Machine Learning*, 13, pages 161--188, 1993.
- [11] G.Szekely, M.L.Padgett, G.Dozier and T.A.Roppel, "Odor Detection using Pulse Coupled Neural Networks", *Int'l Joint Conf. on Neural Networks*, vol. 1, pp. 317-321, 1999.
- [12] R. S. Youssif and C. N. Purdy, "A Multistrategy Signal Pattern Classifier", *MWSCAS*, 2002.
- [13] Courtesy of Electronic Sensor Technology. <http://www.estcal.com/perfumes.html>

The authors thank the State of Ohio/DAGSI program for supporting this research.

Accurate Human Face Extraction using Genetic Algorithm and Subspace Method

Makoto Murakami, Masahide Yoneyama
*Dept. of Information and
Computer Sciences, Toyo University
2100 Nakanodai Kujirai,
Kawagoe Saitama 350-8585, Japan*

Katsuhiko Shirai
*School of Science and Engineering,
Waseda University
3-4-1 Okubo, Shinjuku Tokyo
169-8555, Japan*

Abstract. Subspace method that can express facial images efficiently by linear translation into lower dimensional subspace has wide application for face recognition e.g. identification, facial pose detection etc. In the preprocess of this method the accurate extraction of human face area is required, but it is influenced by the light condition, various background, individual variation and so on, so it has not put into practical use yet. In this paper we examine the subspace method by comparison of the search space, and apply Genetic Algorithm to face extraction and show that the effective results was obtained.

1 Introduction

Lately multimodal interface has been noticed. It is expected as the latest interface that will take the place of the interface using keyboard or mouse. Most of us would accept that human face has important information. To construct the interface using visual information, the recognition of them is indispensable. So a considerable number of studies ever have been made on expression recognition, individual distinction and eye-gaze recognition etc.

Subspace method that can express facial images efficiently by linear translation into lower dimensional subspace has wide application for face recognition e.g. identification, facial pose detection etc. In the preprocess of this method the accurate extraction of human face area is required, but it is influenced by the light condition, various background, individual variation and so on, so it has not put into practical use yet. In this paper, we apply Genetic Algorithm to face extraction and show that the effective results was obtained.

2 Recognition Method

2.1 Subspace Method

2.1.1 Construction of Subspace

Let N features extracted from one image be the following N -dimensional vector x

$$x = [a_1, a_2, \dots, a_N]^T, \quad (1)$$

where a vector x is normalized to $|x| = 1$.

The training set is shown as a set of the vectors

$$\{x_1, x_2, \dots, x_R\}, \quad (2)$$

where R is the step number of rotation.

Then we calculate orthonormal bases in the subspace of the set of vectors.

From the matrix X_{cov}

$$X_{cov} \equiv [x_1 - c, \dots, x_R - c], \quad (3)$$

where c is the average vector of the training set

$$c = \frac{1}{R} \sum_{r=1}^R x_r, \quad (4)$$

we calculate the covariance matrix Q_{cov}

$$Q_{cov} \equiv X_{cov} X_{cov}^T. \quad (5)$$

In the same way from matrix X_{cor}

$$X_{cor} \equiv [x_1, \dots, x_R], \quad (6)$$

we calculate the correlation matrix Q_{cor}

$$Q_{cor} \equiv X_{cor} X_{cor}^T. \quad (7)$$

Then, we solve the following characteristic equation

$$\lambda_j e_j = Q e_j \quad (8)$$

to obtain eigenvalues λ_j and eigenvectors e_j .

k eigenvectors corresponding to bigger eigenvalues

$$(e_1, \dots, e_k), (\lambda_1 \geq \dots \geq \lambda_k \dots \geq \lambda_N) \quad (9)$$

are orthonormal bases of k -dimensional subspace.

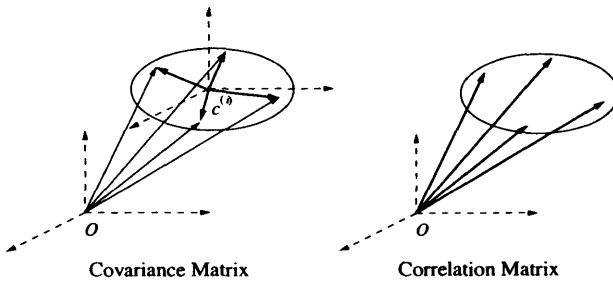


Figure 1: Conceptual Diagram of Subspace

2.1.2 Pose Detection

1. Projection into Subspace

To detect the pose, we use the subspace constructed from the covariance matrix Q_{cov} , because it represents variance from the average, as the conceptual diagram in Fig.1 indicates.

We project the feature vectors extracted from the images into the subspace. The projected vectors p_r

$$p_r = [e_1, e_2, \dots, e_k]^T (x_r - c) \quad (10)$$

are points in the subspace.

2. Pose Detection

A normalized vector y extracted from an input image is projected into the subspace. The projected point z

$$z = [e_1, e_2, \dots, e_k]^T (y - c) \quad (11)$$

is one point in the subspace.

We detect the pose from the distance d_r between the projected point z and the points p_r in the subspace.

$$d_r = |z - p_r| \quad (12)$$

In general, to detect the pose we calculate r that minimizes d_r .

2.1.3 Face Extraction

Projective vector \hat{y} into the subspace of an input vector y is shown as

$$\hat{y} = \sum_{j=1}^k \{(y - c)^T e_j\} e_j \quad (13)$$

when covariance matrix based subspace is used, or

$$\hat{y} = \sum_{j=1}^k \{y^T e_j\} e_j \quad (14)$$

when correlation matrix is used.

We use the squared norm of \hat{y} as the evaluation value of similarity of an input vector to the subspace.

$$\|\hat{y}\|^2 = \sum_{j=1}^k \{(y - c)^T e_j\}^2 \quad (15)$$

$$\|\hat{y}\|^2 = \sum_{j=1}^k \{y^T e_j\}^2. \quad (16)$$

And then, we search the area that maximizes $\|\hat{y}\|^2$ to extract the face area.

2.2 Genetic Algorithm

To search the faces of which both position and size are unknown in an image is just the maximum searching problem of $\|\hat{y}\|^2$ in the 4-dimensional space of (x_c, y_c, M_x, M_y) that are the center of gravity and magnifying power of the extracted area. We apply the Genetic Algorithm to this searching problem.

1. Individual

The genotype $G_k (k = 1, 2, \dots, N)$ of the individual B_k is defined as follows

$$G_k = (x_{ck}, y_{ck}, M_{xk}, M_{yk}), \quad (17)$$

(x_{ck}, y_{ck}) : the center of gravity
 (M_{xk}, M_{yk}) : magnifying power.

2. Fitness Value

The fitness value F is defined as follows

$$F = \|\hat{y}\|^2, \quad (18)$$

where $\|\hat{y}\|^2$ is the evaluation value considered in the section 2.1.3.

3. Selection

Let N be the number of individuals in one generation. the ratio that When $R\%$ individuals of which the fitness values are lower die out, the number is $M = N \times \frac{R}{100}$.

4. Multiplication

$(100 - R)\%$ individuals of which the fitness values are higher survive in the next generation. From this $(N - M)$ individuals new M individuals are generated, and then the total number is fixed in every generation. However, to prevent all individuals from gathering in the same location in the early generation, we make less than $S\%$ individuals survive in the neighbor, however big their fitness values are.

5. Heredity Rule

When from an individual B_i with genotype G_i a new individual $B_{i'}$ is generated, its genotype $G_{i'}$ defined as follows

$$G_{i'} = (x_{ci} + \Delta x_i, y_{ci} + \Delta y_i, M_{xi} + \Delta M_{xi}, M_{yi} + \Delta M_{yi}), \quad (19)$$

Δx_i : random number in $[-\Delta x_{max}, \Delta x_{max}]$

Δy_i : random number in $[-\Delta y_{max}, \Delta y_{max}]$

ΔM_{xi} : random number in $[-\Delta M_{xmax}, \Delta M_{xmax}]$

ΔM_{yi} : random number in $[-\Delta M_{ymax}, \Delta M_{ymax}]$.

$\Delta x_{max}, \Delta y_{max}, \Delta M_{xmax}, \Delta M_{ymax}$ are functions of F_i , determined in the training process as follows

- (a) Let (x_0, y_0) be the coordinate of a face area extracted from an image by the visual observation, and (M_{x0}, M_{y0}) be the magnifying power. We calculate the fitness value F of $(x_0 + \Delta x, y_0 + \Delta y, M_{x0} + \Delta M_x, M_{y0} + \Delta M_y)$ around the face area.
- (b) Then we record the $\Delta x, \Delta y, \Delta M_x, \Delta M_y$ as the functions of fitness value F . When the fitness values are equal, we record the maximum $\Delta x_{max}, \Delta y_{max}, \Delta M_{x_{max}}, \Delta M_{y_{max}}$. And when some fitness values can't appear in this process, we define $\Delta x_{max}, \Delta y_{max}, \Delta M_{x_{max}}, \Delta M_{y_{max}}$ as follows

$$\Delta x_{max} = x_{max}(F - 1)^2 \quad (20)$$

$$\Delta y_{max} = y_{max}(F - 1)^2 \quad (21)$$

$$\Delta M_{x_{max}} = M_{x_{max}}(F - 1)^2 \quad (22)$$

$$\Delta M_{y_{max}} = M_{y_{max}}(F - 1)^2, \quad (23)$$

where $x_{max}, y_{max}, M_{x_{max}}, M_{y_{max}}$ is the maximum of $\Delta x_{max}, \Delta y_{max}, \Delta M_{x_{max}}, \Delta M_{y_{max}}$ in the neighbor of face area respectively.

6. Mutation

M individuals are generated at random once in every L generations, not following the heredity rule defined above. This can prevent from convergence in the local maximum to lead the optimum solution.

3 Experiment

3.1 Outline

In this experiment we use face images taken from 36 horizontal directions as training data. The size of them is normalized to 64×64 pixels. Then we construct the subspace from edge and smoothing feature vector as Fig.2 shows, to scale back the influence of the light condition, individual variation and so on.

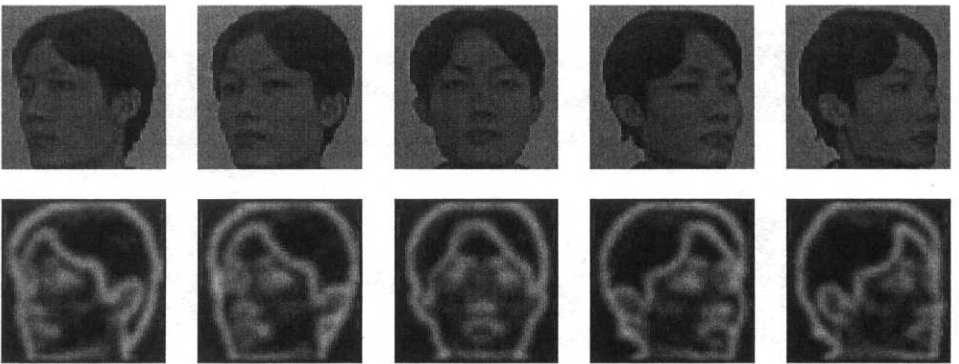


Figure 2: Training Data

Preparatory for the experiment of human face extraction, we compare the search spaces of $\|\hat{y}\|^2$ considered in the section 2.1.3 in two methods (using covariance matrix and correlation

matrix), and examine if it is able to reject the background area. Then we apply the Genetic Algorithm considered in the section 2.2 to human face extraction.

3.2 Results and Considerations

3.2.1 Comparison of Search Spaces

To extract the face area is just to reject the background area. But the various background images are not trained, so the curved surface in the search space is unpredictable. Fig.3 indicates noticeable examples of the search space on the assumption that face size is fixed for the images taken on the various conditions. The vertical axis in the graphs indicates the $\|\hat{y}\|^2$ defined in the section 2.1.3, so extracting the face area is equal to looking for the maximum in this space. The black region in the graphs shows the face area with which the template overlaps more than half.

As the graphs indicate, in the method using correlation matrix the score in the background is not restrained. So it is possible to converge on the points in the background. On the other hand, in the covariance matrix based method the score in the face area obviously higher than in the background though the graph is discontinuous curved space. Therefore in the next section we experiment with the method using covariance matrix.

Then we compare the inner product using some bases with all bases, to consider the number of bases constructing subspace. The pie charts in Fig.4 indicate the rates of bases of which inner product is maximized in the face area and the background of various images respectively. These graphs show that the important bases are e_1, e_2, e_3 , and e_2, e_3 are unresponsive to the background, so possibly restrain the score in the background. Fig.3 shows also search spaces of the sum of the inner product of e_1, e_2, e_3 . There is not a great difference between using all bases and using three bases (e_1, e_2, e_3) in search space itself. So we use e_1, e_2, e_3 in the experiment in the next section.

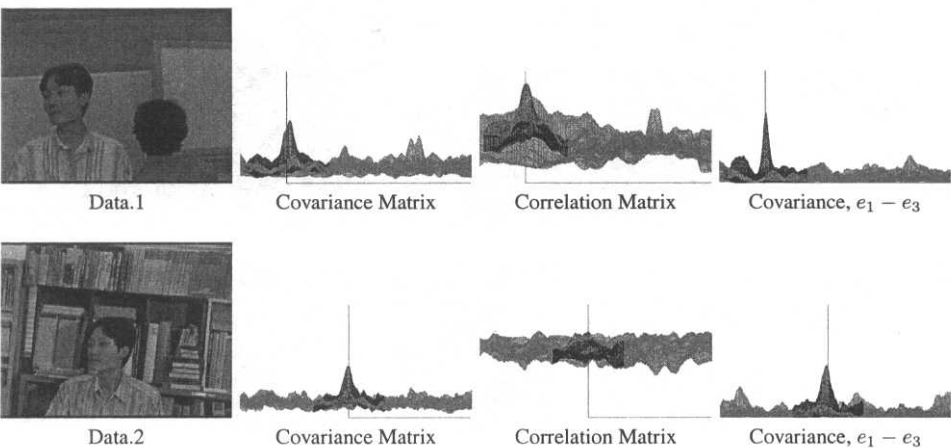


Figure 3: Search Space

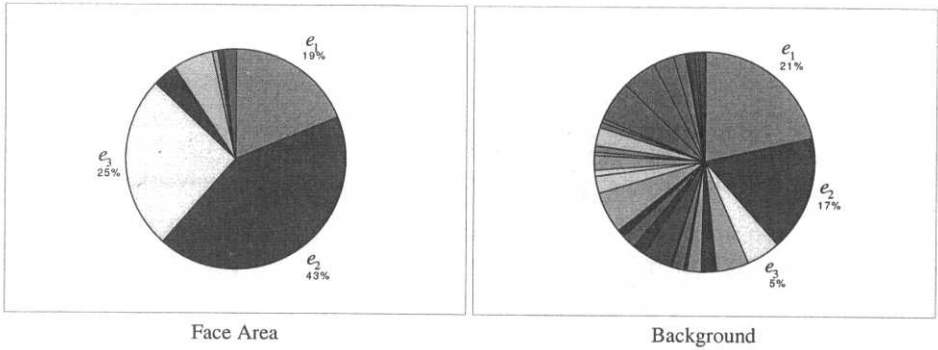


Figure 4: Rate of Bases of which inner product is maximized

3.2.2 Face Extraction using Genetic Algorithm

We use the Genetic Algorithm considered in the section 2.2 to extract face area. The number of individuals in one generation is 24, 12 individuals of which the fitness values are lower die out, and then from the survived 12 individuals new 12 individuals are generated. The new 12 individuals are mutated once in every 5 generations. Furthermore to prevent all individuals from gathering in the same location, we make not more than 3 individuals searching in the neighbor survive in the next generation. Therefore, four or more region can be searched in at the same time.

Then we consider the evaluation value of extraction accuracy. Let C be the area extracted by the visual observation, and H be the area extracted by GA. We calculate the size S_D of the area D defined as

$$D = (C \cap \bar{H}) \cup (\bar{C} \cap H). \quad (24)$$

And we define an evaluation value R as

$$R = \frac{S_D}{S_C}. \quad (25)$$

Then, we define another evaluation value as the differences D_r between the correct pose and the pose r minimizes d_r defined in the section 2.1.2. In this experiment we use 250 images with various backgrounds.

The results are shown in Fig.5-Fig.7. The horizontal axis in the graph indicates the generation of GA. The vertical axis in the upper part of the graph indicates fitness value F , and it in the lower part indicates accuracy R .

Fig.5 shows an example to converge in early generation; face area has been extracted in about the 24th generation. Fig.6 shows an example to converge getting out of local solutions; individuals begin gathering around the face area in the 64th generation, and the face area get extracted accurately by and by. Fig.7 shows an example of failure; no individual is generated near the face area during the search process.

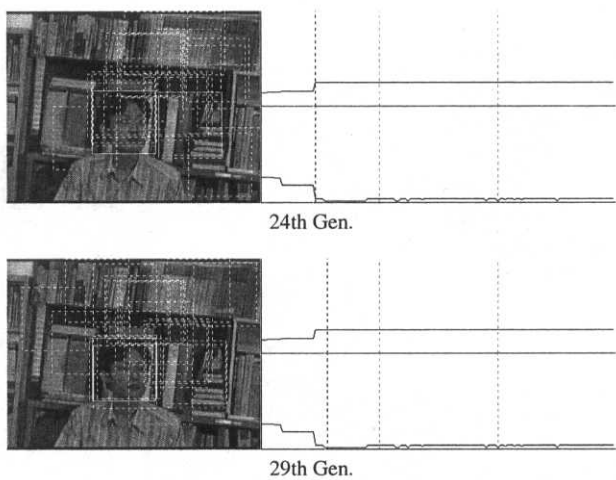


Figure 5: Result 1 (success)

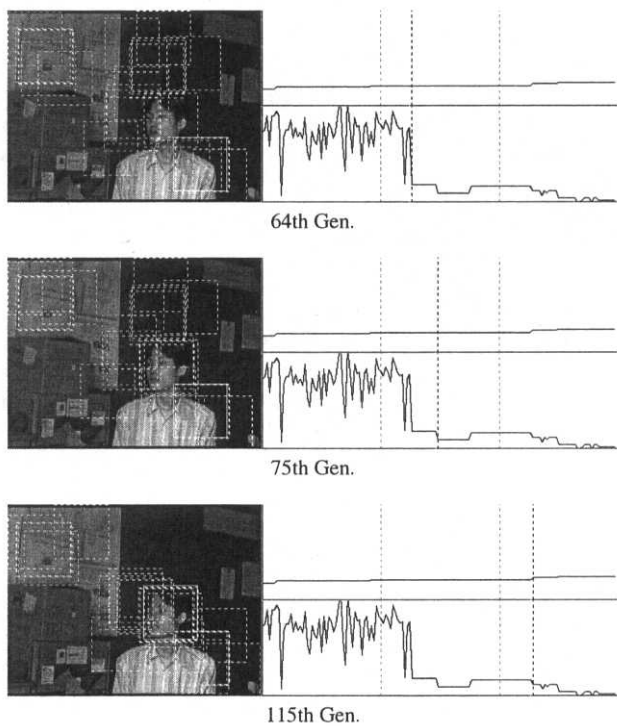


Figure 6: Result 2 (success)

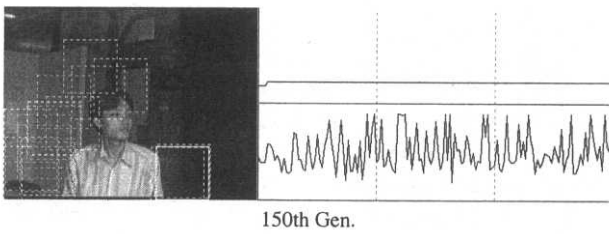


Figure 7: Result 3 (failure)

Then we consider the change of accuracy in generations. Fig.8 indicates the extraction accuracy. The horizontal axis in these graphs shows R , the vertical axis shows the number of the extracted images of which the accuracy is R . And the line graph shows the accumulated rate. Here, one step of the horizontal axis is 0.05. Fig.9 indicates the extraction accuracy by another evaluation value D_r . The horizontal axis in these graphs shows $|D_r|$.

As generations go by, the accuracy of both R and D_r are gradually improved. It follows that our genetic algorithm is effective for the discontinuous search space.

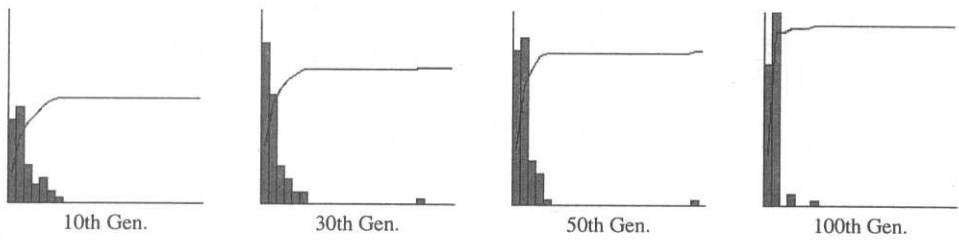


Figure 8: Extraction Accuracy 1

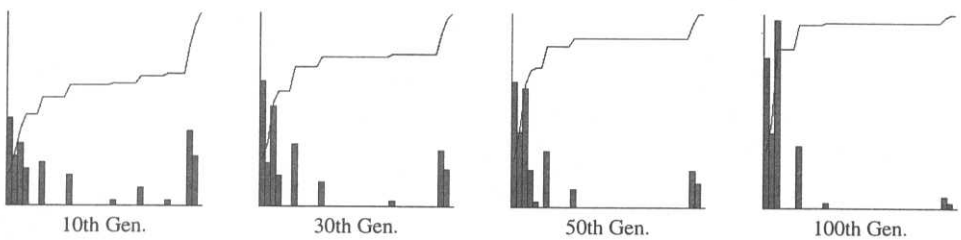


Figure 9: Extraction Accuracy 2

4 Conclusion

In the method of constructing subspace using covariance matrix the score in the face area obviously higher than in the background, this method is effective to reject the background. But the search space is discontinuous curved space. So we apply Genetic Algorithm to this search space and the effective results was obtained.

Our method will be well fitted to the problem of face recognition. Subspace method or eigenface method is proposed for the identification or verification in the first place. It is important to note that the rejection rate for the unknown is very important for these applications such as security systems. Furthermore we are investigating the issues of robustness to changes in lighting, head orientation and the number of people.

For applications such as human interface, it is necessary to reduce computational effort, because of the processes for the image sequence. But once face location is fixed in a frame, we can track the face region for the next frames. Therefore we are investigating the accurate and robust tracking method for applications.

References

- [1] Erkki Oja: "Subspace Methods of Pattern Recognition," Research Studies Press, England, 1983.
- [2] Yoshie Komatsu and Yasuo Ariki: "Orientation Invariant Face Extraction and Recognition based on Subspace Method," Technical Report of IEICE, PRU95-191, pp7-14, 1996-01.
- [3] Yoshiaki Sugiyama and Yasuo Ariki: "Facial Region Tracking and Training Method by Subspace Projection," Technical Report of IEICE, PRMU97-162, pp77-82, 1997-11.
- [4] Tomoharu Nagao, Takeshi Agui and Hiroshi Nagahashi: "Pattern Matching of Binary Shapes Using a Genetic Method," Transaction of IEICE, J76-D-II, No.3, pp557-565, 1993-03.
- [5] Akira Hara and Tomoharu Nagao: "Extraction of facial regions of arbitrary directions from still images with a genetic algorithm," Technical Report of IEICE, HCS97-12, pp37-44, 1997-09.
- [6] Shinya Masunaga and Tomoharu Nagao: "Extraction of human facial regions in still images using a genetic algorithm," Technical Report of IEICE, PRMU95-160, pp13-18, 1995-11.

The Evolutionary Learning Rule for System Identification in Adaptive Finite Impulse Filters

Oscar MONTIEL¹, Oscar CASTILLO², Patricia MELIN², Roberto SEPULVEDA¹

¹CITEDI, National Polytechnic Institute, Tijuana, Mexico.

²Dept. of Computer Science, Tijuana Institute of Technology, Tijuana, Mexico

Abstract. In this paper, we are proposing an approach for integrating evolutionary computation applied to the problem of system identification in the well-known statistical signal processing theory. Here, some mathematical expressions are developed in order to justify the learning rule in the adaptive process when a Breeder Genetic Algorithm is used as the optimization technique. In this work, we are including an analysis of errors, energy measures, and stability.

1. Introduction

The problem of determining a mathematical model for an unknown system by observing its input-output data pairs is known as system identification, and it is an important step when we wish to design a control law for a specific system. Real systems are non-linear and have time variations hence the best control laws that we can obtain are those based using real time data from continuous-time stochastic processes [1].

Traditionally, system identification has been performed in two ways:

1. Using analytic models, i.e., obtaining mathematically the transfer function.
2. Using experimental input-output data. In this way, the identification can be achieved in two forms: non-parametric and parametric.

In this paper we are interested in parametric models. As we mentioned, there are several and well-known techniques to perform the system identification process, most of the parametric techniques are gradient guided and are limited in highly multidimensional search spaces. The system identification process generally involves two top-down steps, and these are: structure identification, and parameter identification. In the first step, we need to apply *a priori* knowledge about the target system for determining a class of model within the search for the most suitable model is going to be conducted [2] [3].

Here, we are using an evolutionary algorithm known as Breeder Genetic Algorithm (BGA) that lays somehow in between Genetic Algorithms and Evolutionary Strategies (ESs). Both methods usually start with a randomly generated population of individuals, which evolves over the time in a quest to get better solutions for a specific problem. GAs are coded in binary forming strings called chromosomes, they produce offsprings by sexual reproduction. Sexual reproduction is achieved when two strings (i.e. parents) are recombined (i.e. crossover), generally the parents are selected stochastically, the search process is mainly driven by the recombination operation, and the mutation is used as secondary search operator with low probability, to explore new regions of the search space. An ES is a random search, which models natural evolution by asexual reproduction [4]. It uses direct representation, that is, a gene is a decision variable and its allele is the value of the variable [5], in ES the mutation is used as the search operator, and uses the (μ, λ) strategy as a selection method. Thus, the BGA can be seen as a combination of ESs and GAs, because it handles direct real variables, and *truncation selection*, which is very similar to the strategy (μ, λ) , and the search process is mainly driven by recombination making BGAs similar to GAs [6] [7] [8] [9] [10].

2. The Generic Identification Problem

Figure 1 shows the generic problem of system identification. Here, we have a digital signal input $x(n)$ that is fed to the unknown system and to the adaptive filter at the same time, in this figure there is a "black box" enclosed by dashed lines, its output is called the desired response signal and it is represented by $d(n)$, the adaptive system will compute a corresponding output signal sample $y(n)$ at time n . Both signals, $d(n)$ and $y(n)$ are compared subtracting the two samples at time n , to obtain a desired response signal, this concept is expressed in equation (1)

$$e(n) = d(n) - y(n) \quad (1)$$

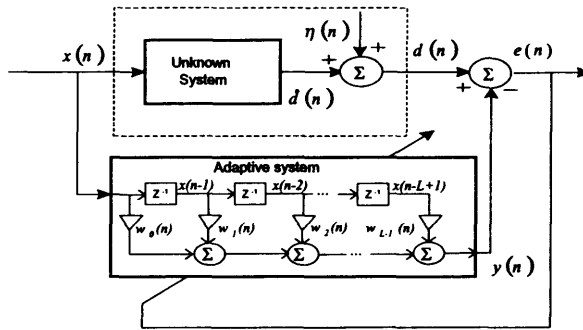


Figure 1. System identification with noise presence. The adaptive system uses an adaptive FIR, also known as transversal filter. In this figure z^{-1} , represents the unit delay element and each $w_i(n)$ is a multiplicative gain within the system.

This block might have a pole-zero transfer function, an all-pole or auto-regressive transfer function fixed or time varying, a non-linear mapping, or some other complex system. In the dashed "black box", we have an additive noisy signal known as the observation noise signal because it corrupts the observation of the signal at the output of the unknown system [17], thus the real desired signal $\hat{d}(n)$ is contaminated with noise; hence the signal $d(n)$ is given by equation (2)

$$d(n) = \hat{d}(n) + \eta(n) \quad (2)$$

In the adaptive system block we could have any system with a finite number of parameters that affect how $y(n)$ is computed from $x(n)$. In this work, we are using an adaptive filter with a Finite Impulse Response (FIR filter), and it is represented by the equation

$$y(n) = \sum_{i=0}^{L-1} w_i(n) x(n-i) \quad (3)$$

or in vectorial form

$$y(n) = W^T(n) X(n) = X^T(n) W(n) \quad (4)$$

where the coefficient vector $W(n)$ is

$$W(n) = [w_0(n) \ w_1(n) \ \dots \ w_{L-1}(n)] \quad (5)$$

here $\{w_i(n)\}$ $0 \leq i \leq L-1$ are the L parameters of the system at time n . The input signal vector is given in vectorial form by,

$$X(n) = [x(n) \ x(n-1) \ \dots \ x(n-L+1)] \quad (6)$$

In the system identification problem, the adaptive filter has the task of represent accurately the signal $d(n)$ at its output, this is the case when $y(n) = \hat{d}(n)$. In this stage, the filter has made its work identifying the portion of the unknown system driven by $x(n)$, but

frequently this is an ideal goal, because if we are using a linear FIR filter, then the unknown system should be a filter of the same structure. In real world problems, the identification is successful if we meet with some criterion in the error value. Moreover, in real world problems, the output of the unknown system $\hat{d}(n)$ is contaminated with noise $\eta(n)$.

Generally, we do not have direct access to the uncorrupted output $\hat{d}(n)$ of the plant, instead we have a noisy measurement of it, in this case the output is given by equation (7). Then, we can say that the adaptive filter has reached the optimum if we find a value $y(n)$; $d(n)$ this is achieved when we find an optimum weight vector's parameter $W(n)$,

$$W(n) = W_{opt}(n) \quad (7)$$

There are several methods to obtain the optimum values, each of them has its own advantages and limitations, but they share the same limitation when the search space is big and multi-modal because they are gradient-guided local search techniques.

The correction term of the learning rule usually is function of the signal input $x(n)$, the desired output $d(n)$, and the old weight estimate values $w_i(n-1)$. Thus, we can write the compact expression, as follows [11] [12]:

$$W(n) = W(n-1) + f[X(n), d(n), W(n-1)] \quad (8)$$

Following these concepts, the least-mean-square-algorithm (LMS) can be written as shown in equation (9)

$$W(n) = W(n-1) + \mu \cdot X(n) \cdot [e(n)] \quad (9)$$

or, expanding the expression for the error, as

$$W(n) = W(n-1) + \mu \cdot X(n) \cdot [d(i) - X_i^T \cdot W(n-1)] \quad (10)$$

where μ is the step-size parameter.

In order to analyze the learning rule for the BGA in a formal way, we are going to introduce the next notation. We will use $\mathcal{P}_r(n)$, to indicate the actual population of r individuals of coefficient vectors $W_r(n)$. By example, one individual is represented as,

$$W_0(n) = [w_{0,0}(n) \ w_{0,1}(n) \ K \ w_{0,L-1}(n)] \quad (11)$$

where $w_{0,1}(n)$, belongs to the coefficient 1 of the individual number 0. $W'_r(n)$ belongs to the next generation of coefficient after the evolutionary process. Then we have

$$\tilde{W}_r(n) = \begin{bmatrix} W_0(n) \\ W_1(n) \\ \vdots \\ W_r(n) \end{bmatrix} = \begin{bmatrix} w_{0,0}(n) & w_{0,1}(n) & K & w_{0,L-1}(n) \\ w_{1,0}(n) & w_{1,1}(n) & K & w_{1,L-1}(n) \\ M & M & M & M \\ w_{r,0}(n) & w_{r,1}(n) & L & w_{r,L-1}(n) \end{bmatrix} \quad (12)$$

In the evolutionary process, the learning rule is a several steps process. The evolutionary learning process begins with the evaluation of every coefficient vector of the population represented in equation (12) by $\mathcal{P}_r(n)$, this process is done using a specific evaluation function " f_i " known as the fitness function. After the whole population has been evaluated, we need to perform a selection of the best of this population, generally truncation selection is a good choice, with this process we obtain a pool $\mathcal{P}_r^*(n)$. The whole process, from expression (13) to (16), is repeated until we reached the optimum value.

$$\mathcal{P}_r^* = T\%(\text{feval}(\mathcal{P}_r(n))) \quad (13)$$

$$W'_r(n) = W_r(n) + f[W_r(n), W_r(n)] \quad (14)$$

$$\mathcal{P}_r^*(n) = \Delta(\mathcal{P}_r^*) \quad (15)$$

$$\mathcal{P}_r(n) = \mathcal{P}_r^*(n) \quad (16)$$

The fitness function (f_i) will guide the search to find the optimum.

3. The Breeder Genetic Algorithm

In order to adjust the parameter vector, we used a BGA, which was designed according to the methods and theories used in the science of livestock breeding [11], and is based on advanced statistical methods [12][13][14][15][16][17].

The BGA is defined as an eight-tuple

$$BGA = (P_g^0, N, T, \Gamma, \Delta, HC, F, term) \quad (17)$$

where P_g^0 is the initial population of size N , T is the truncation threshold commonly referred as $T\%$, Γ represents the recombination operator, Δ is the mutation operator, HC is a hill climbing method (by example: the gradient guided LMS algorithm), F is the fitness function and $term$ is the termination criterion. In the BGA, $T\% \frac{P}{100}$ best individuals at each generation are selected and mated randomly, until the number of offsprings is equal the size of the population. A basic scheme of a BGA is shown is described in [5][14].

Referring the BGA procedure to our problem, we need to generate randomly a population with P individuals, where each individual consists of n variables, each of them related with a parameter, i.e. filter's coefficient. By example, an individual might consist of 50 floating point variables, in this case, the adaptive FIR that we need to use will have 50 coefficients. The whole population is evaluated using a fitness function, specifically designed to measure the aptitude of each individual. This evaluation is done applying a specific signal $x(n)$ (figure 1) to both systems (unknown and adaptive systems) in order to calculate an error signal obtained from the output of the unknown and adaptive systems. This error signal is the core of the fitness function. After each individual of the parent population was evaluated, the best individual should be inserted in the population of the next generation, $P'(t)$. In order to obtain this new population, which will replace the parent population, the BGA uses truncation selection. In truncation selection a percentage of the best individuals are selected to mate, obtaining offsprings, self-mating is prohibited [15].

The mutation operator is applied to each offspring, and the resulting individuals are inserted in the new population $P'(t)$. The process is repeated until a termination criterion is met. There are several recombination operators, but in this work, we used Extended Intermediate Recombination (EIR), in order to use this operator, we have: if $x = (x_1, \dots, x_n)$ and $y = (y_1, \dots, y_n)$ are the parents, then the successor $z = (z_1, \dots, z_n)$ is calculated by:

$$z_i = x_i + \alpha_i (y_i - x_i) \quad i = 1, K, n \quad (18)$$

where $\alpha_i \in [-d, 1+d]$ is chosen with uniform probability for each i and $d \geq 0$, a good choice is $d=0.25$, which is the one that we used. The goal of the mutation operator is to modify one or more parameters of z_i , the modified objects (i.e., the offsprings) appear in the landscape within a certain distance of the unmodified objects (i.e., the parents). In this way, an offspring z' , where $z' = (z'_1, \dots, z'_n)$ is given by:

$$z'_i = z_i \pm range_i \cdot \delta \quad (19)$$

where $range_i$ defines the mutation range, and is calculated as $(\lambda \cdot searchinterval_i)$. In the Discrete Mutation operator (DM) λ is normally set to 0.1 or 0.2 and is very efficient in some functions[4], but also we can set λ to 1. This is the case of the Broad Mutation Operator (BM) proposed in [16] to solve problems where the distance between the optimums and the actual position is larger than the DM operator could reach. This is the case of Scwefel's function [16]. We will have in one step, that the mutation operator can reach points from z_i only within the distance given by $\pm range_i \cdot \delta$. The sign + or - is chosen with probability of 0.5. The variable δ is computed by

$$\delta = \sum_{i=0}^{K-1} \varphi_i 2^{-i} \quad \varphi_i \in 0,1 \quad (20)$$

Before mutation we set each φ_i equals to 0, then each α_i is mutated to 1 with probability $p_\delta = 1/k$, and only $\alpha_i = 1$ contributes to the sum. On the average there will be just one α_i with value 1, say α_j . Then δ is given by

$$\delta = 2^{-j} \quad (21)$$

In formula (20), K is a parameter originally related to the machine precision, i.e. the number of bits used to represent a real variable in the machine we are working with, traditionally K used values of 8 and 16. In practice, however, the value of K is related to the expected value of mutation steps, in other words, the higher K is, the more fine-grained is the resultant mutation operator [5].

4. Energy and Error Analysis

The objective of an adaptive scheme is to provide a better model for the unknown system, this is done using the output error sequence given in equation (1), this error sequence measures how far $d(i)$ is from $y(i)$, in order to update the coefficient vector $W(n)$, providing a better model at each iteration. When the adaptive system is trying to match the unknown system, there exist different sources of errors. The first source is due to the mathematical model that we are using to identify the unknown system. By example, if the unknown plant has the model of the Infinite Impulse Response Filter (IIR) of equation (22), it is evident that we would need an infinitely long tapped-delay line, for which is practically impossible its realization. Therefore, when we are using a finite tapped delay line the modeling errors are inevitable.

$$y(n) = \sum_{i=0}^{\infty} v_i(n)x(n-i) \quad (22)$$

In this work, we used as an unknown system an IIR filter specified by the second-order difference equation given by (23). This problem was taken from [20].

$$d(n) - d(n-1) + 0.9d(n-2) = x(n) \quad \forall n \quad (23)$$

The adaptive system is a FIR filter with 50 coefficients, as the filter has a higher degree of freedom, the input-output response characteristic of the adaptive model will converge to match closely those of the unknown system.

If we have solved the system identification problem, and the model that we are using is a good one, then we should have a way to measure the parameter errors in the identification process [21]. If we considered that $V(n)$ represents the coefficient vector of the unknown system represented by equation (22), then we have to measure how far is $W(n)$ from $V(n)$, this quantity ($W_e(n)$) will be referred as the weight error at time n . The weight vector $W(n)_{OPT}$ is the optimal vector that better will represent $V(n)$.

$$W_e(n) = W(n)_{OPT} - W(n) \quad (24)$$

Using equation (25) we can obtain the value known as *a priori estimation error* (e_a), which measure how far is $X_i^T W(n-1)$ from the uncorrupted output term $X_i^T W_{OPT}$, hence we have

$$e_a(i) = X_i^T W_e(n-1) \quad (25)$$

If we use the most recent weight vector, we can define the *a posteriori estimation error* (e_p),

$$e_p(n) = X^T(n)W_e(n) \quad (26)$$

A good way to quantify the effect of these errors is the energy and power measures. Then the energy E of a signal $x(n)$ is [17].

$$E \equiv \sum_{n=-\infty}^{\infty} |x(n)|^2 \quad (27)$$

The energy of a signal can be finite or infinite. The signal $x(n)$ is called an energy signal if E is finite, thus is $0 < E < \infty$.

We can define the signal energy of the sequence $x(n)$ over the finite interval $-0 \leq n \leq N$ as [16]

$$E_N = \sum_{n=0}^{N-1} |x(n)|^2 \quad (28)$$

then we can express the signal energy as [16]

$$E \equiv \lim_{N \rightarrow \infty} E_N \quad (29)$$

The average power of the signal $x(n)$ is defined as [16],

$$P \equiv \lim_{N \rightarrow \infty} \frac{1}{N} E_N \quad (30)$$

We will use the definition of power provided in (16), to calculate a short-term average *ASE* (Average of Squared Error), which will serve us in two ways; one is to estimate the convergence rate of the evolutionary algorithm, and the second way is for using the inverse of this function ($1/ASE$) as the aptitude function, which will guide the evolutionary algorithm to get the optimum coefficients. The *ASE* function is defined as

$$ASE(m) = \frac{1}{K} \sum_{k=n+1}^{n+K} e^2(k) \quad (31)$$

Here, $m=n/K=1,2,\dots$. The recommended averaging interval K may be selected to be (approximately) $K=10L$ [16].

5. Stability Analysis

In any practical application, an analysis of system's stability is important, since an unstable system usually exhibit erratic and extreme behavior and cause overflow [21]. To prove stability in this work, we used the next theorem, as well as concepts about causal systems:

Theorem. *An arbitrary relaxed system is said to be bounded input \square bounded output (BIBO) stable if and only if every bounded input produces a bounded output.*

Considering the previous theorem, if $x(n)$ is the bounded input, there must be a constant M_x such that

$$|x(n)| \leq M_x < \infty \quad (32)$$

and, if the output is bounded, there must be a constant M_y such that

$$|y(n)| < M_y < \infty \quad (33)$$

Although that we are working with time varying system, for analysis purpose we are considering a window time where the system can be treated as a Linear Time invariant system (LTI), and via mathematical induction generalize this result. Then, if we take the absolute value of both sides of equation (3), we obtain

$$|y(n)| = \left| \sum_{i=0}^{L-1} w_i(n) x(n-i) \right| \quad (34)$$

Then, we have

$$|y(n)| \leq \sum_{i=0}^{L-1} |w_i(n)| |x(n-i)| \quad (35)$$

If the input is bounded, there exists a finite number M_x which satisfies equation (32), i. e.

$|x(n)| \leq M_x$ then we can rewrite equation (35), as

$$|y(n)| \leq M_x \sum_{i=0}^{L-1} |w_i(n)| \quad (36)$$

From equation (36), we observe that the output is bounded if the impulse response of the system is absolutely summable, i.e.

$$\sum_{i=0}^{L-1} |w_i(k)| < \infty \quad (37)$$

In this work, equation (36) is easily satisfied since we have a finite number of coefficients $w_i(k)$, and their values also are finite. This result implies that any excitation of a finite duration, applied to the input of the system must produce an output of nature "transient", this is when the system is stable, the input amplitude will decay eventually with time. This

condition implies that $W(z)$ must contain the unit circle within its Region of Convergence (ROC). Then,

$$W(z) = \sum_{i=0}^{L-1} w_i(k) z^{-n} \quad (38)$$

in consequence,

$$|H(z)| \leq \sum_{i=0}^L |w_i(k) z^{-n}| = \sum_{n=-\infty}^{\infty} |w_i(k)| |z^{-n}| \quad (39)$$

evaluating on the unit circle (i.e. $|z|=1$),

$$|H(z)| \leq \sum_{n=-\infty}^{\infty} |h(n)| \quad (40)$$

Then, if the system is BIBO stable, the unit circle is contained in the ROC of $H(z)$. From equation(3), we can see that the adaptive system is causal because its output at time n depends only on present and past inputs, but does not depend on future inputs. A LTI system is causal if and only if the ROC of the system function is the exterior of a circle of radius $r < \infty$, including the point $z = \infty$.

6. Experimental Results

This algorithm was realized using the procedure BGA explained in section 3. We selected as the adaptive system a FIR filter (ARMA model), We used as unknown system an IIR filter specified by the second-order difference equation given in (23), this problem was taken from [20]. A signal sequence called $x(n)$ is used as the input to the unknown system and the adaptive FIR filter, then this sequence is the training signal. Using the appropriate training signal it is very important to be successful in the identification of a system. A good signal should contain at least 10 frequency components. For this reason random signals like white noise and gaussian white noise are generally used. In this work, we used both signal to train the system, and both are good enough to find optimized parameters. With these signals, we trained the adaptive FIR. In both cases, the tests were made with a population of 300 individuals. In the EIR operator, $d = 0.25$. We used the DM operator with $\lambda = 0.1$, and $\delta = 16$. In order to speed up the processing time we use only 80 samples to calculate ASE, we choose this number experimentally. We ran the BGA for 100 generations.

An important point about the development of this algorithm is: the fitness function f_i represents the value of the i -th individual, then

$$f_i = \frac{1}{ASE_i(m)} \quad (41)$$

The training white noise signal $x(n)$, was generated using the Matlab command $x = \text{rand}(1, 100) * 2 - 1$; in figure 2 we show an histogram of this signal

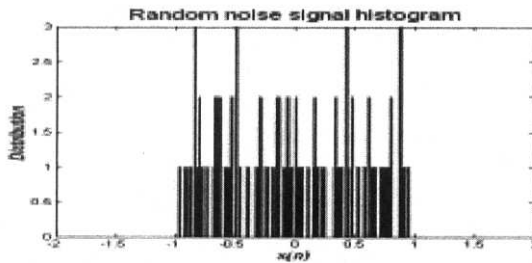


Figure 2. For training the ARMA model we generated a sequence of 100 samples of white noise at each generation.

The training gaussian white noise signal $x(n)$, was generated using the Matlab command $x = \text{wgn}(100, 1, -6)$; in figure 3 we show an histogram of this signal.

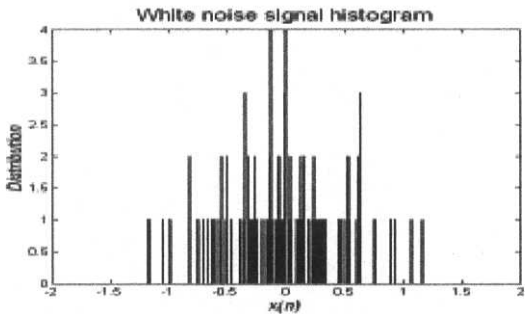


Figure 3. Gaussian distribution of 100 samples.

For all the experiments, we analyzed the system response using Bode plots. In the Bode plots we used normalized frequency expressed in radians per sample (rad/samp). The normalized frequency f is calculated using equation (42), where F is the frequency that we want to normalize and F_s is the sampling frequency.

$$f = \frac{F}{F_s} \tag{42}$$

Hence, if we are sampling at $F_s=1000$ rad/samp , and the value in the Bode plot is $f=10^0$, then the frequency F is $F=fF_s=10^0 \cdot 1000=1000$ rad/samp. Although, we must fulfill the sampling theorem of equation (43)

$$F_s \geq 2F_{\max} \tag{43}$$

Experiment #1.

In this experiment, the adaptive filter has 50 coefficients. We used a white noise signal sequence for training the adaptive system. The maximal fitness value found was 481.0087, at generation 96, this is shown in Figure 4. Then $ASE=1/481.0087 \approx 0.0020$. In figure 5, we have three graphics, the upper one is the desired output ($d(n)$) of the unknown system, the one in the middle belong to the output of the adaptive system; and the graphic at the bottom corresponds to the signal error, this graphic represents the value difference at each sample. Figure 6 belongs to the unit step response, and the interpretation is similar to figure 5. Figure 7 and 8 are the Bode plots, here we can analyze the system response in magnitude and phase. In both figures we are using normalized frequency expressed in rad/seg.

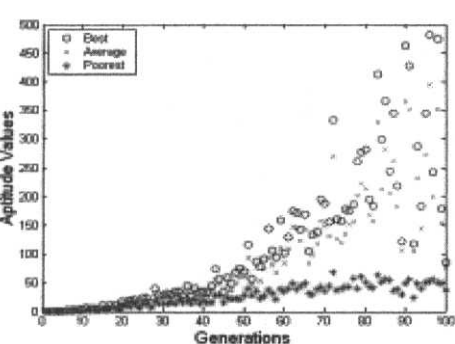


Figure 4. Fitness graphic, the \circ belongs to the best fitness found at each generation, as well as the x belongs to an average fitness value of the generation, and the symbol $*$ is for the poorest fitness.

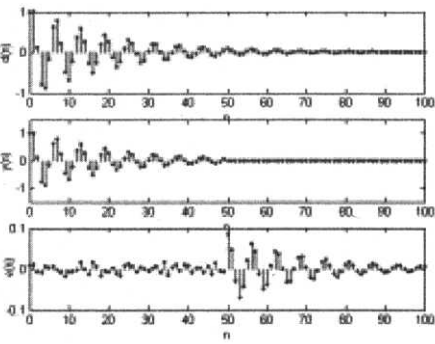


Figure 5. Unit impulse response. Experiment 1.

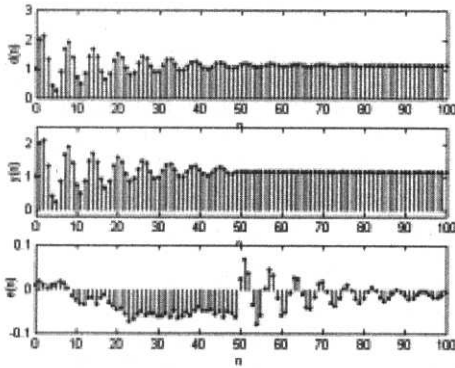


Figure 6. Unit Step response. Experiment 1.

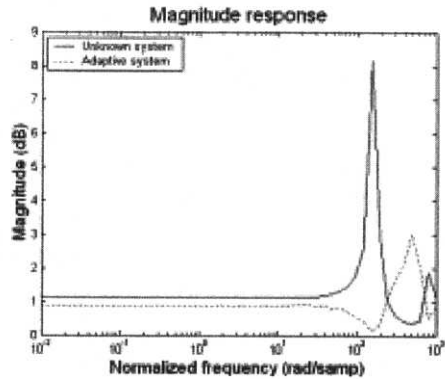


Figure 7. Behavior of the adaptive system.

Experiment #2.

In this experiment, the adaptive filter has 50 coefficients. The maximal fitness value found was 695.8930, at generation 95. Then $ASE=1/695.8930 \approx 0.00144$.

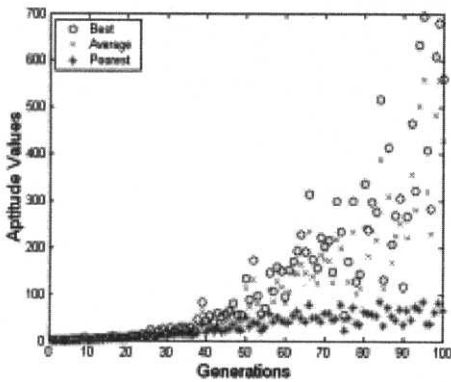


Figure 8. Fitness function's values. Experiment 2.

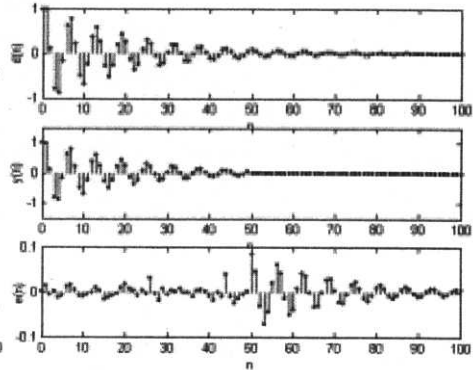


Figure 9. Unit impulse response.

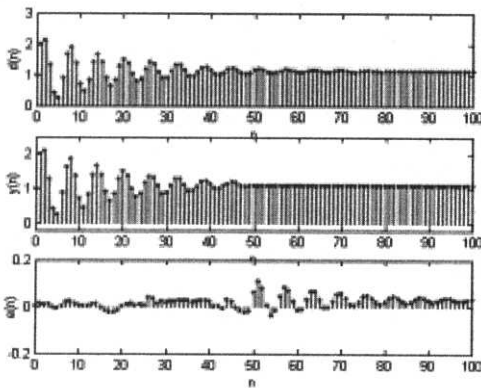


Figure 10. Unit step response. Experiment 2.

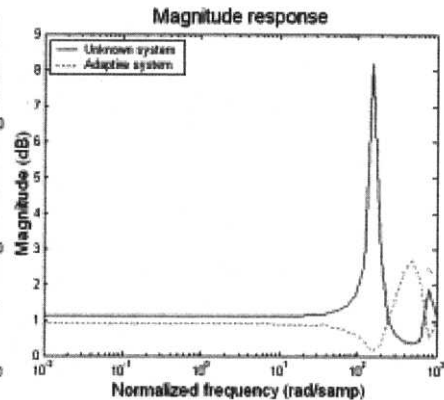


Figure 11. Magnitude response. Experiment 2.

7. Conclusions

In this paper we developed a set of equations in order to explain the learning process of the adaptive system when we are using a BGA in the system identification problem. We realized several experiments; here we showed the result of four of them. In all cases, a full optimization was performed obtaining low rates of average error and good magnitude and phase response. Obviously, when we used more coefficients in the ARMA model we got a lower average error, this is natural, because as more degree of freedom has the ARMA model (FIR filter), the input-output response characteristic of the adaptive model will converge to match closely the unknown system.

8. References

- [1] Carl W. Helstrom. Probability and stochastic processes for engineers. Helstrom U.S.A., 1991.
- [2] J.-S. R. Jang, C.-T. Sun, E. Mizutani. "Neuro-Fuzzy and Soft Computing, A Computational Approach to Learning and Machine Intelligence". Matlab Curriculum Series. Prentice Hall. 1997.
- [3] Harris, S. P. & Ifeakor, E. C. "Nonlinear FIR Filter Design by Genetic Algorithm", in "1st Online Conference on Soft Computing", 1996. <http://www.lania.mx/~ccoeillo/EM00/harrisabs.txt>
- [4] H. Mühlenbein. "6 Genetic Algorithms". Borneo <http://citeseer.nj.nec.com/181734.html>
- [5] Lluís A. Belanche. "A Study in Function Optimization with the Breeder Genetic Algorithm". Research Report LSI-99-36-R. Universitat Politècnica de Catalunya, 1999. <http://citeseer.nj.nec.com/belanche99study.html>
- [6] De Falco, A. Della Cioppa, P. Natale and E. Tarantino. "Artificial Neural Networks Optimization by means of Evolutionary Algorithms", 1997. <http://citeseer.nj.nec.com/3503.html>
- [7] White Michel S., Flockton J. Stuart., "Adaptive recursive filtering using evolutionary algorithms", <http://citeseer.nj.nec.com/392797.html>
- [8] D. M. Etter, M. J. Hicks, and K. H. Cho. "Recursive filter design using an adaptive genetic algorithm". In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 82), volume 2, pp. 635-638, IEEE, 1982.
- [9] Oscar Montiel, Roberto Sepúlveda, Oscar Castillo, Patricia Melin. "Application of a Breeder Genetic Algorithm for Filter Optimization", In Proceedings of the International Conference on Artificial Intelligence IC-AI'2001, volume III, pp. 1199-1205, Las Vegas, Nevada, USA. 2001.
- [10] Oscar Montiel, Roberto Sepúlveda, Oscar Castillo, Patricia Melin. "A Breeder Genetic Algorithm for Filter Optimization". In World Multiconference on Systemics, Cybernetics and Informatics (ISAS/SCI 2001), volume IX Industrial Systems: Part I, pp. 556-562, Orlando, Florida, USA. 2001.
- [11] Vijay K. Madisetti, Douglas B. Williams. "The Digital Signal Processing Handbook", A CRC Handbook Published in Cooperation with IEEE Press, pp. 18-1, 18-7, 20-1, 20-4. 1997.
- [12] Montiel Oscar H., Castillo Oscar, Melin Patricia, Sepúlveda Roberto. "The Evolutionary Learning Rule in System Identification". The 6th. World Multiconference on Systemics, Cybergenics and Informatics (SCI 2002), volume XVII Industrial Systems and Engineering III. pp. 205-210.
- [13] H. Mühlenbein. "The equation for the response to selection and its use for prediction". <http://citeseer.nj.nec.com/181734.html>
- [14] Heinz Mühlenbein, Dirk Schlierkamp-Voosen. "The science of breeding and its application to the breeder genetic algorithm BGA". Evolutionary Computation, 1(4):335-360, 1994.
- [15] H. Mühlenbein, J. Bendisch, H.-M. Voigt. "From recombination of genes to the estimation of distribution II. Continuous parameters". Parallel Problem Solving from Nature, Springer: Berlin, pp: 188-197, 1996.
- [16] Ivanoe De Falco. "Sample Contributed Book". John Wiley & Sons. Editor Jenny Smith. 1997. <http://citeseer.nj.nec.com/cs>
- [17] Heiz Mühlenbein and Schlierkamp-Voosen. "Predictive Model for Breeder Genetic Algorithm". Evolutionary Computation. 1(1): 25-49, 1993.
- [18] Ralf Salomon. "Genetic Algorithms and the $O(n \ln n)$ Complexity on Selected Test Functions". <http://citesser.nj.nec.com/109962.html>
- [19] Vijay K. Madisetti, Douglas B. Williams. "The Digital Signal Processing Handbook", A CRC Handbook Published in Cooperation with IEEE Press, pp. 18-1, 18-7, 20-1, 20-4. 1997.
- [20] Vinay K. Ingle & John G. Proakis. "Digital Signal Processing using Matlab". Ed. Brooks/Cole, 2000, pp. 371-377.
- [21] Oscar Montiel, , Oscar Castillo, Roberto Sepúlveda. "Error and Energy Measures in System Identification Using an Evolutionary Algorithm", Proceedings of the International Conference on Artificial Intelligence IC-AI'2002. Las Vegas, Nevada, USA. 2002.

2D-Histogram Lookup for Low-contrast Fault Processing

Mario Köppen, Raul Vicente Garcia, Xiufen Liu and Bertram Nickolay
Fraunhofer IPK, Department Pattern Recognition
Pascalstr. 8-9, 10587 Berlin, Germany
email: {mario.koeppen|raul|xiufen.liu|nickolay}@ipk.fhg.de

Abstract. This paper presents a framework for low-contrast texture fault processing based on 2D-Histogram lookup. 2D-Histogram lookup is a variant of 2D-Lookup operation. For 2D-Lookup, in two differently processed versions of the same image, grayvalue pairs from the same location in both images are replaced by the corresponding entry in a given two-dimensional matrix. The two operations and the matrix have to be provided for full algorithm specification. For 2D-Histogram lookup, the matrix is derived from the 2D-Histogram of both processed images. The main advantage of using the 2D-Histogram is the darkening of rarely occurring structures in the image, while highly probable image structures becomes bright. The so-processed images are then given to a 2D-Lookup procedure for automatic filter generation. For low-contrast texture faults, i.e. faults which are hard to separate from the background texture, the approach shows better performance in fault region detection than the approach of 2D-Lookup adaptation without 2D-Histogram lookup. For handwriting separation from textured background, no achievement was obtained.

1 Introduction

Recently, a framework was presented for the automated generation of texture filters [1]. The framework was based on the adaptation of 2D-Lookup algorithm application of Mathematical Morphology to a user-given task (see section 2.2 for the definition of this algorithm). The task includes the provision of the iconic description of the image foreground structure (texture fault, handwriting on textured background) by means of a so-called goal image, and the original image.

The framework has shown a good performance on a broad range of filtering task, with filtering here means the separation of image foreground and background.

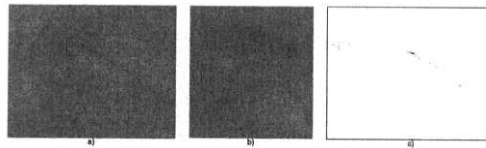


Figure 1: Low contrast stroke processing: (a) shows a stroke on a collagen sheet, (b) a detail enlarged. (c) gives the output of best 2D-Lookup adaptation, demonstrating the poor performance of this algorithm in this case.

However, it also showed some falacities. So, a rather poor performance on foreground appearance with very low contrast against the background was noted. Fig. 1 gives an example. The stroke-like fault structure in subfigure (a) (a scratch in a collagen sheet), clearly visible for a human, becomes nearly "invisible" while getting enlarged (subfigure (b)). Subfigure (c) shows the result of 2D-Lookup adaptation, revealing only some random dots of the fault.

Analyzing this problem, it came out that the framework only employed a purely local image processing. Local processing here means that the domain of the image processing operations is bounded to a spatial neighborhood of each pixel. Detection of a stroke as the one in fig. 1 can not be achieved by such a local processing.

This paper presents an approach to solving such problems, by introducing an extension of the 2D-Lookup framework, referred to as 2D-Histogram lookup procedure. This extension can be considered a pre-processing with the whole image as operation domain. It is also based on the 2D-Lookup, but uses the normalized 2D-Histogram as lookup matrix. This gives higher contrasting of locally "untypical" image structures, thus simplifying the consecuting 2D-Lookup adaptation task.

Section 2 describes the now extended framework and its components, while section 3 gives some examples for the approach. The paper concludes with a summary and the reference.

2 General framework

2.1 Overview

Figure 2 shows the general framework for 2D-Lookup adaptation, with the optional 2D-Histogram lookup extension. The components will be described in the following subsections. As can be seen, the extension has nearly the same structure as the framework without extension, only the specification of the lookup matrix LT is different.

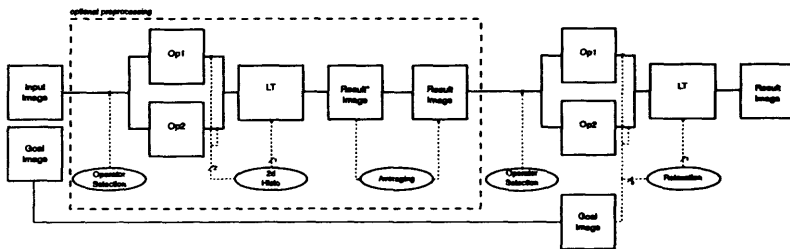


Figure 2: Extended framework for 2D-Lookup adaptation, with optional 2D-Histogram lookup pre-processing module.

The 2D-Lookup algorithm is specified by three components: image operator 1, image operator 2, giving operation images Op1 and Op2, and the lookup matrix LT. Both image operators can be chosen arbitrarily, hence the high adaptive potential of this algorithm. The choice of the operators is performed by the operator selection module.

For choosing an appropriate lookup matrix, the framework without extension and the extension follow different strategies. While in the framework the lookup matrix is set up in accordance with the goal image, in order to achieve maximum similarity of 2D-Lookup result

and goal image (measured with a fitness function), in the extension the lookup matrix is taken from the 2D-Histogram of two processed versions of the input image. However, in this case, a post-processing of the result image by an averaging operation is necessary, as indicated by the naming "result*."

2.2 2D-Lookup

The 2D-Lookup algorithm stems from mathematical morphology [4], [5]. It was primarily intended for the segmentation of color image channels. However, the algorithm can be generalized to be used for grayvalue images as well.

For starting off the 2D-Lookup algorithm, the two operation images 1 and 2, which are of equal size, need to be provided. The 2D-Lookup algorithm goes over all common positions of the two operation images. For each position, the two pixel values at this position in operation images 1 and 2 are used as indices for looking-up the 2D-Lookup matrix. The matrix element, which is found there, is used as pixel value for this position of the result image. If the matrix is bi-valued (e.g. has only entries being either 0 for Black or 255 for White), the resultant image is a binary image.

Let I_1 and I_2 be two grayvalue images, defined by their image functions g_1 and g_2 over their common domain $P \subseteq \mathcal{N} \times \mathcal{N}$:

$$\begin{aligned} g_1 : P &\rightarrow \{0, \dots, g_{max}\} \\ g_2 : P &\rightarrow \{0, \dots, g_{max}\} \end{aligned} \quad (1)$$

The 2D-Lookup matrix is also given as an image function l , but its domain is not the set of all image positions but the set of tupels of possible grayvalue pairs $\{0, \dots, g_{max}\} \times \{0, \dots, g_{max}\}$. Then, the resultant image function is given by:

$$\begin{aligned} r : P &\rightarrow S \\ r(x, y) &= l(g_1(x, y), g_2(x, y)). \end{aligned} \quad (2)$$

In standard applications, every grayvalue is coded by eight bit, resulting in a maximum grayvalue of 255. Also, the domain of the image function is a rectangle. Since the goal image is supplied as a binary one and in order to keep user instruction as simple as possible, in the following the 2D-Lookup matrix contains only binary entries Black (0) and White (255).

2.3 2D-Histogram-based 2D-Lookup

The 2D-Histogram lookup is the 2D-Lookup algorithm with using the normalized 2D-Histogram as lookup matrix. The 2D-Histogram is defined as a matrix $H(g_1, g_2)$, derived from two operation images I_1 and I_2 of same size and resolution. An entry counts the number of occurrences of the same grayvalue pair at the same position (x, y) in both images. Thus

$$H(g_1, g_2) = \sum_{(x,y) \in I_1 \cap I_2} \delta(I_1(x, y), g_1) \delta(I_2(x, y), g_2) \quad (3)$$

with $\delta(a, b)$ being 1 for $a = b$ and 0 otherwise.

For using the 2D-Histogram as lookup matrix, its entries have to be scaled into the range of feasible grayvalues. For usual goal image sizes, entries in the 2D-Histogram are seldom

larger than the maximum grayvalue, so the entries itself can be used as lookup matrix values, bounded at 255. Also, a contrast improvement by linearization is reasonable, for gaining better contrast in the result image. Linearization is the operation of making the sum histogram of grayvalues linear. It replaces each grayvalue in the image with the relative number of grayvalues equal or below this grayvalue. Hereby, the large number of Zero entries in the 2D-Histogram is neglected, thus starting linearization at grayvalue 1.

While gaining higher contrast by linearization, the number of different grayvalues now in the result image becomes reduced. This gives images with a cluttered appearance of the 2D-Lookup matrices in the following 2D-Lookup adaptation step. This effect can be reduced by using a Gaussian smoothing operator of size 3 on the linearized image.

2.4 Generic operator design

The operator selection is based on a tree-like representation of the image processing operations. Each node in the tree refers to a generic image processing operations out of the set Squaring, Square Root, Move, Ordered Weighted Averaging (OWA [7]), fuzzy T-Norm, and Fuzzy Integral [6] for nodes of arity one, and pixel-wise addition, subtraction, multiplication and T-Norm for nodes of arity two (higher nodes are not used). The parameters of such a node operation are given by a resource, with the same structure for all nodes. This parameter structure resource indicates an offset vector (for move operation), a weighted mask (for OWA, T-Norm and Fuzzy Integral), a weighting vector (for OWA) and some flags and mode values. See [1] for more details on the operation specifications.

In the framework presented here, the operator selection is performed four times, two for the extension to get the operation images, from which the 2D-Lookup matrix is derived, and two operations to get the operation images, for which the 2D-Lookup matrix is adapted for the goal image. In all four cases, the operation trees and the parameter structures are randomly initialized, and adapted later on by Genetic Programming [2] [3] as optimization procedure to gain high fitness values.

2.5 Operator fitness

In order to compare the output image of the 2D-Lookup with the goal image, a quality function has to be designed for the comparison of two binary images. For such a comparison, two sets are given, the reference set of the goal image and the pattern set of the result image.

In the following, *countBlack* is the number of black pixels in the result image ($B + C$), *matchBlack* is the number of black pixels of the result image, which are also black in the goal image (B), and *refBlack* is the number of black pixels of the goal image ($A + B$). Then, the following ratios can be computed:

$$r_1 = \frac{\text{matchBlack}}{\text{refBlack}} \quad (4)$$

is the amount of reference pixels matched by the pattern,

$$r_2 = 1.0 - \frac{\text{countBlack} - \text{matchBlack}}{N - \text{refBlack}} \quad (5)$$

is the amount of correct white pixels set in the result image (N is the total number of image pixels), and

$$r_3 = \frac{\text{matchBlack}}{\text{countBlack}} \quad (6)$$

is the amount of matching pixels of the result image. The multiple objective here is to increase these measures simultaneously. After performing some experiments with the framework, it was decided to use the following weighted sum of these three objectives as fitness measure:

$$f = 0.1r_1 + 0.5r_2 + 0.4r_3 \quad (7)$$

This fitness measure has the following properties: (1) It counts better for subsets of the reference. Subsets obtain a fitness value of at least 0.9, since r_2 and r_3 are 1 in this case; (2) It counts better for subsets of the reference, which are supersets of other subsets of the reference; (3) A white image gives a fitness of 0.5, therewith refusing to assign a good fitness value to the empty subset of the reference.

These properties make this fitness measure useful for genetic search. A genetic search evolves its population towards the higher weighted objective first. In our case this means, that measure r_2 , weighted with 0.5, is evolved first. In other words, the first subgoal of the genetic search is to allocate as many correct white positions as possible. Due to the weighting of 0.4 for the r_3 -part, the search then tries to allocate correct black positions of the reference, while the correct white allocations persist in the pattern. Once the pattern is reduced to a subset of the reference, the only way to increase the fitness is to expand the subset towards the whole reference set. This begins, when the fitness exceeds a value of about 0.9.

In the following, it is assumed, that the two operation images and the goal image are given. The question is how to derive a suitable 2D-Lookup matrix, which gives the best match between goal image and result image by the fitness measure given in the last subsection. The interesting point here is, that this derivation can be done by reusing this fitness measure. To prove this, assume a 2D-Lookup matrix, where all but one positions are set to White (1), and only a single position (g_1, g_2) is set to Black (0). Then, the 2D-Lookup will give a resultant image with all positions (x, y) set to Black, for which operation 1 yielded pixel value g_1 and operation 2 yielded pixel value g_2 . Usually, there will be only a few black pixels within the result image. Now, the fitness measure will give values above 0.9, if the set of black pixels lies completely within the reference, no matter, how many pixels are there. So, a criterion can be given for setting a pixel to Black or White in the 2D-Lookup matrix.

Let $l_{(x,y)}$ be a 2D-Lookup matrix constituted by setting only the pixel at (x, y) to Black, and $r_{(x,y)}$ be the result of the 2D-Lookup with the operation images 1 and 2 and this 2D-Lookup matrix. Then

$$l(x, y) = \begin{cases} \text{Black,} & \text{if } f(r_{(x,y)}) > 0.88 \\ \text{White} & \text{otherwise.} \end{cases}$$

In case there are no black pixels in $r_{(x,y)}$ at all, $l(x, y)$ is set to Gray, which stands for positions within the 2D-Lookup matrix, whose pixel value pairs do never occur within the operation images 1 and 2 at the same location. The value 0.88 has been chosen instead of 0.9 to give the adaptation some tolerance.

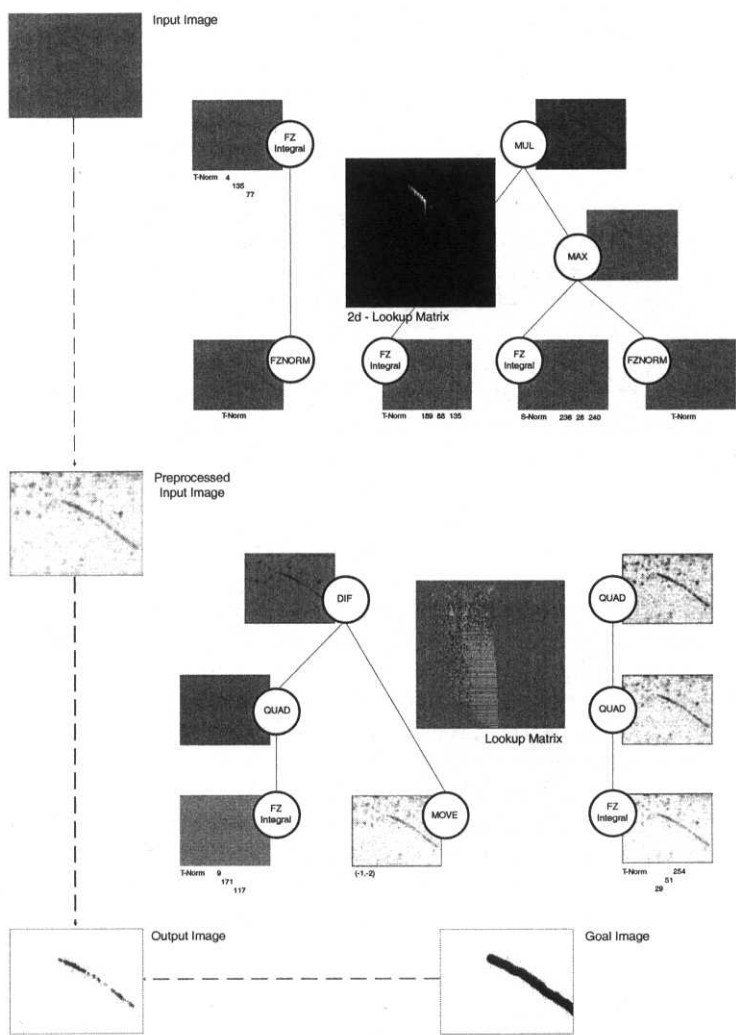


Figure 3: Complete example for the extended framework adapted to the low-contrast stroke in fig. 1.

3 Examples

The framework for 2D-Lookup adaptation was extended in order to enable the processing of low-contrast foregrounds. Figure 3 gives a complete example for such an adapted filtering. This example demonstrates several things:

- The final result image in the lower left is much more similar to the goal image than in fig. 1(c).
- The result of the 2D-Histogram lookup preprocessing step has increased the contrast of the stroke against the background. This allows for the following 2D-Lookup adaptation step to achieve that better performance.
- The operation trees are not "smart" in the sense that parts of it may do not have much influence on the operation result of the full tree. So, the left wing of the operation 1 of the 2D-Lookup (the tree left below in fig. 3) basically produces a gray image, from which a shifted version of the preprocessed image is subtracted. Such parts can be removed in a re-design phase of the filters after adaptation by hand (especially when they involve computationally more expensive node operations like the Fuzzy Integral).
- The 2D-Lookup for this example is basically a lookup with the co-occurrence matrix of the preprocessed image.
- The essential part of this filter lies in the operation 2 of the preprocessing (upper right tree): the 2D-Histogram, with its line-like structure, forks for some higher grayvalue pairs, thus stimulating a separation of image parts into foreground and background.

Figure 4 shows another examples, without the intermediate results. This is a so-called "vibrating knife" fault that may occur in collagen slicing. The extended framework shows a good performance here as well.

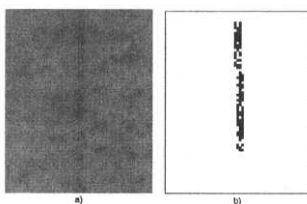


Figure 4: Result of the extended framework for vibrating-knife fault.

Another application field for the extended framework could be handwriting separation on textured background. Figure 5 shows an example for this. However, obviously the performance here is worse than for the framework without extension (subfigure (b) without preprocessing, (c) with preprocessing). The reason lies in the preprocessed image (d), which is not related to the goal image at all: parts of handwriting have been removed, since their local appearance is typical for the whole image, especially when handwriting stroke direction fits the direction of the lines in the background pattern. Thus, the consecuting 2D-Lookup adaptation is not able to recover those parts from the preprocessed image.

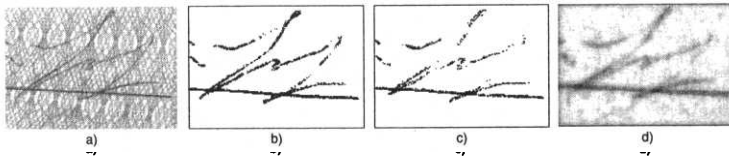


Figure 5: Extended framework applied to handwriting separation shows lower performance.

4 Summary

This paper presented a framework for low-contrast texture fault processing, i.e. faults which are hard to separate from the background texture. The framework is a two-stage one, based on 2D-Histogram lookup and consecuting 2D-Lookup adaptation. 2D-Histogram lookup itself is a variant of 2D-Lookup operation. For 2D-Lookup, in two differently processed versions of the same image, grayvalue pairs from the same location in both images are replaced ("looked-up") by the corresponding entry in a given two-dimensional matrix. The two operations and the matrix have to be provided for full algorithm specification. This gives the algorithm the 2D-Lookup algorithm high versatility for the application in various contexts. For 2D-Histogram lookup, the matrix is derived from the 2D-Histogram of both processed images. The main advantage of using the 2D-Histogram is the darkening of rarely occurring structures in the image, while highly probable image structures becomes bright. The so-processed images are then given to a 2D-Lookup procedure for automatic filter generation, employing Genetic Programming for the generation of appropriate image processing operations. For low-contrast texture faults, the approach shows better performance in fault region detection than the approach of 2D-Lookup adaptation without 2D-Histogram lookup. Performance in case of handwriting was not so good due to common appearance of patterns in both, the handwriting and the background.

References

- [1] Mario Köppen and Bertram Nickolay. Genetic programming based texture filtering framework. In Nikhil R. Pal, editor, *Pattern Recognition in Soft Computing Paradigm*, FLSI Soft Computing Series - Vol. 2, pages 275–304. World Scientific, 2001.
- [2] John R. Koza. *Genetic Programming – On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, London MA, 1992.
- [3] John R. Koza. *Genetic Programming II – Automatic Discovery of Reusable Programs*. MIT Press, Cambridge, London MA, 1994.
- [4] Jean Serra. *Image Analysis and Mathematical Morphology*. Academic Press, London, 1982.
- [5] Jean Serra. *Image Analysis and Mathematical Morphology 2: Theoretical Advances*. Academic Press, London, 1988.
- [6] M. Sugeno. *Fuzzy Control*. Nikkan Kogyo Shimbun-sha, 1988. (in Japanese).
- [7] R.R. Yager. On ordered weighted averaging aggregation operators in multi-criteria decision making. *IEEE Transaction on System, Man & Cybernetics*, 18:183–190, 1988.

Section 13

Industrial Applications

This page intentionally left blank

A Railway Interlocking Safety Verification System Based on Abductive Paraconsistent Logic Programming

Kazumi Nakamatsu¹, Jair M. Abe², Atsuyuki Suzuki³

¹`nakamatu@hept.himeji-tech.ac.jp`

*School of H.E.P.T., Himeji Institute of Technology,
1-1-12 Shinzaike, HIMEJI 670-0092 JAPAN*

²`jairabe@uol.com.br`

*Department of Informatics, ICET, Paulista University,
Rua Dr.Bacelar, 1212 04026-002 Sao Paulo BRAZIL*

³`suzuki@cs.inf.shizuoka.ac.jp`

*Faculty of Information, Shizuoka University
3-5-1 Johoku, HAMAMATSU 432-8011 JAPAN*

Abstract. We introduce a safety verification system for railway interlocking based on a paraconsistent logic program called an EVALP and its abductive procedure called SLDNFA ϵ , which is an extended version of SLDNFA by Denecker and De Schreye. The original idea of the safety verification method was proposed by Morley. Based on his idea, we have already proposed a safety verification system for railway interlocking based on EVALP. However, if the request to be verified contains wrong data, the system cannot detect the error. Therefore, we extend the safety verification system to reason the correct data by EVALP programming and its abduction.

1 Introduction

Safety verification for railway interlocking is a crucial issue and automated safety verification systems are required. The first automated safety verification system for railway interlocking was introduced by Dr. Morley in his Ph.d thesis [6] as far as we know. His safety assurance method is ; establish the safety properties for railway interlocking ; logically verify whether requests to secure or release railway routes by signal operators contradict the safety properties or not. He used a higher order language HOL [4] as the verification tool. However, his system has incompleteness due to higher order logic, that is, it has formulas that cannot be proved even if they are theorems.

In order to overcome the incompleteness, referring to Morley's method, we have proposed an automated safety verification system for railway interlocking [11] based on a paraconsistent logic program called an Extended Vector Annotated Logic Program (EVALP for short) [9, 10], which can deal with defeasible deontic reasoning.

The basic method of the EVALP based safety verification is that : the safety properties, route securing requests and sub-route release requests can be represented by using deontic notions such as obligation and forbiddance in EVALP ; therefore, the safety properties are translated into EVALP clauses and stored in the database ; each request is divided into if-part and then-part, and the if-part is added to the database as EVALP

clauses representing facts ; then, the then-part in EVALP clauses representing permission is verified whether it contradicts the database or not by EVALP programming. In the verification process, if the then-part is not verified and the answer **no** is returned from the EVALP, we have the question which part of the if-part is wrong ? Then, we want to know the correct if-part. In the new system with abduction, the correct if-part can be reasoned by an abductive procedure for EVALP called SLDNFA ϵ .

Abduction is a reasoning by which the cause is inferred from the observed result, that is, a reasoning by which the explanation for the observed fact can be inferred. In fact, SLDNFA by Denecker and De Schreye is known to be an abductive procedure. SLDNFA ϵ that is an extended abductive procedure for EVALP is used in our system, although we do not explain about SLDNFA ϵ in details in this paper.

This paper is organized as follows : after EVAL being reviewed, the abductive procedure for EVALP, SLDNFA ϵ is introduced with a simple example ; next, an outline of the safety verification system based on abductive procedure is described ; with taking an railway interlocking example from Morley's Ph.d thesis, the safety verification process is shown in details.

2 Preliminary

In this section, we introduce the abductive procedure SLDNFA ϵ after reviewing EVALP briefly [10, 9]. We assume that the reader is familiar with the usual notions of ordinary first order logic and logic programming in Lloyd[5].

2.1 EVALP

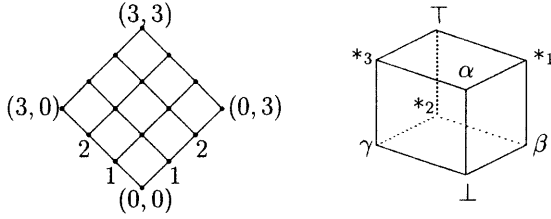
Generally, in annotated logic programs [1], a truth value called an *annotation* is explicitly attached to each literal. For example, let p be a literal, μ an annotation, then $p : \mu$ is called an *annotated* literal. A partially ordered relation is defined on the set of annotations that has a complete lattice structure. An annotation in EVALP called an *extended vector annotation* has a form of $[(i, j), \mu]$. Let us take an extended vector annotated literal $p : [(i, j), \mu]$. The first component (i, j) is a 2-dimentional vector annotation whose first component i indicates the degree of positive information(true) to support the literal p and the second component j indicates the degree of negative information(false) to support the literal p . The third component μ is an index that denotes notions such as fact(α), obligation(β) non-obligation(γ) and so on. The complete lattice \mathcal{T} of extended vector annotations is defined as follows : $\mathcal{T} = \mathcal{T}_v \times \mathcal{T}_d$,

$$\mathcal{T}_v = \{(i, j) \mid 0 \leq i \leq m, 0 \leq j \leq m\} \quad \text{and} \quad \mathcal{T}_d = \{\perp, \alpha, \beta, \gamma, *_1, *_2, *_3, \top\},$$

where i, j and m are non-negative integers. The ordering of the lattice \mathcal{T}_v is denoted in the usual fashion by a symbol \preceq_v and defined as : let $\vec{v}_1 = (x_1, y_1)$ and $\vec{v}_2 = (x_2, y_2)$,

$$\vec{v}_1 = (x_1, y_1) \preceq_v \vec{v}_2 = (x_2, y_2) \quad \text{iff} \quad x_1 \geq x_2 \quad \text{and} \quad y_1 \geq y_2.$$

The ordering of \mathcal{T}_d is denoted by a symbol \preceq_d and the complete lattices \mathcal{T}_v and \mathcal{T}_d are described by the Hasse's diagrams in Figure 1. The intuitive meanings of the members of \mathcal{T}_d are : \perp (unknown), α (fact), β (obligation), γ (non-obligation), $*_1$ (both fact and obligation), $*_2$ (both obligation and non-obligation), $*_3$ (both fact and non-obligation) and \top (inconsistent). The Hasse's diagram(cube) shows that the lattice \mathcal{T}_d is a tri-lattice in which the direction $\vec{\gamma\beta}$ indicates *deontic truth*, the direction $\vec{\perp *_2}$ indicates the

Figure 1: Lattice \mathcal{T}_v and Lattice \mathcal{T}_d

amount of *deontic knowledge* and the direction $\overrightarrow{\perp\alpha}$ indicates *factuality*. Therefore, for example, the annotation β can be intuitively interpreted to be deontically truer than the annotation γ and the annotations \perp and $*_2$ are deontically neutral, i.e., neither obligation nor not-obligation. The ordering of \mathcal{T} is denoted by a symbol \preceq and defined as : let $[(i_1, j_1), \mu_1]$ and $[(i_2, j_2), \mu_2]$ be extended vector annotations,

$$[(i_1, j_1), \mu_1] \preceq [(i_2, j_2), \mu_2] \text{ iff } (i_1, j_1) \preceq_v (i_2, j_2) \text{ and } \mu_1 \preceq_d \mu_2.$$

In general, annotated logic has two kinds of negation, an epistemic negation and an ontological(strong) negation, however, EVALP does not contain the ontological negation. In general, the epistemic negation followed by an annotated literal is defined as a mapping between annotations. There are two kinds of epistemic negation, \neg_1 and \neg_2 , in EVALP, which are defined as mappings over \mathcal{T}_v and \mathcal{T}_d , respectively.

Definition 1 (Epistemic Negations in EVALP, \neg_1 and \neg_2)

$$\begin{aligned} \neg_1([(i, j), \mu]) &= [(j, i), \mu], & \forall \mu \in \mathcal{T}_d \\ \neg_2([(i, j), \perp]) &= [(i, j), \perp], & \neg_2([(i, j), \alpha]) = [(i, j), \alpha], \\ \neg_2([(i, j), \beta]) &= [(i, j), \gamma], & \neg_2([(i, j), \gamma]) = [(i, j), \beta], \\ \neg_2([(i, j), *_1]) &= [(i, j), *_3], & \neg_2([(i, j), *_2]) = [(i, j), *_2], \\ \neg_2([(i, j), *_3]) &= [(i, j), *_1], & \neg_2([(i, j), \top]) = [(i, j), \top]. \end{aligned}$$

The epistemic negations \neg_1 and \neg_2 followed by extended vector annotated literals can be eliminated by the above syntactic operations in **Definition 1**.

Deontic notions are represented by extended vector annotations. For example, an extended vector annotated literal $p: [(3, 0), \alpha]$ is intuitively interpreted as “the literal p is known to be a fact of strength 3”, and $q: [(0, 2), \beta]$ is also done as “the literal q is known to be forbidden of strength 2”.

Definition 2 (well extended vector annotated literal) Let p be a literal.

$$p: [(i, 0), \mu] \quad \text{or} \quad p: [(0, j), \mu]$$

are called *well extended vector annotated literals*, where i, j and m are non-negative integers ($1 \leq i, j \leq m$) and $\mu \in \{ \alpha, \beta, \gamma \}$.

Definition 3 (EVALP) If L_0, \dots, L_n are well extended vector annotated literals,

$$L_1 \wedge \dots \wedge L_n \rightarrow L_0$$

is called an *extended vector annotated logic program clause* (EVALP clause for short). An *Extended Vector Annotated Logic Program* is a finite set of EVALP clauses.

Example 1 Suppose an EVALP

$$P = \{ \quad p: [(3, 0), \alpha], \quad (1)$$

$$p: [(2, 0), \alpha] \rightarrow q: [(0, 3), \beta], \quad (2)$$

$$q: [(0, 2), \gamma] \rightarrow r: [(3, 0), \beta] \quad \}. \quad (3)$$

Since $[(2, 0), \alpha] \preceq [(3, 0), \alpha]$, from (1) and (2), the extended vector annotated literal

$$q: [(0, 3), \beta] \quad (4)$$

is derived. However, since $[(0, 2), \gamma] \not\preceq [(0, 3), \beta]$, from (3) and (4), the extended vector annotated literal $r: [(3, 0), \beta]$ cannot be derived.

2.2 Abductive EVALP

In our safety verification system, we use the abductive procedure $SLDNF_{\epsilon}$ that is an extended version of the abductive procedure $SLDNFA$ [3]. Abduction is a reasoning to find some explanations for the observed result, and some of *abducible predicates* that are previously provided as the candidates for the explanations are abducted. We describe abductive EVALP programming with taking a simple example.

Example 2 We consider the following EVALP :

$$P = \{ \quad p: [(2, 0), \alpha], \quad (5)$$

$$q: [(2, 0), \alpha] \rightarrow r: [(2, 0), \beta], \quad (6)$$

$$s: [(0, 2), \alpha] \wedge t: [(2, 0), \alpha] \rightarrow r: [(1, 0), \beta], \quad (7)$$

$$u: [(2, 0), \alpha] \wedge p: [(1, 0), \alpha] \rightarrow s: [(0, 3), \alpha] \quad \}, \quad (8)$$

and the following abducible predicates $\{ q, t, u \}$. Then, if we inquire

$$r: [(2, 0), \beta], \quad (9)$$

from the EVALP clause (6), $q: [(2, 0), \alpha]$ is abducted as an explanation for the query (9). In addition, if we inquire

$$r: [(1, 0), \beta], \quad (10)$$

from the EVALP clause (6), $q: [(2, 0), \alpha]$ is abducted, and from the EVALP clauses (7) and (8), $t: [(2, 0), \alpha]$, and $u: [(2, 0), \alpha]$ are abducted as explanations for the query (10).

3 Safety Verification Based on Abductive EVALP

3.1 Outline of the Safety Verification

We introduce an outline of the abductive EVALP based safety verification system for railway interlocking. The verification is carried out as shown in Figure 2 :

- (1) the safety properties for railway interlocking, which must be assured when interlocking, are translated into EVALP clauses and stored in the system ; Abducible predicates that represent points, track section etc. are also stored in the system as preprocessing ;

- (2) requests to be verified, that are, route security requests called Panel Route Requests (PRRs for short) or route release requests called Sub-Route Release (SRRs for short), are translated into EVALP clauses ; those requests have a form of 'if **A** then **B**', and the if-part **A** and the then-part **B** are translated in EVALP clauses that represent fact and permission, respectively ;
- (3) the translated if-part **A** is input to the system and the translated then-part **B** is inquired from the system as logic programming inquiry ; then if **yes** is returned from the system, the safety of the request is assured, if **no** is returned, by SLDNFA ϵ (abductive procedure for EVALP), the system abduces the correct if-part **A'** in EVALP clauses.

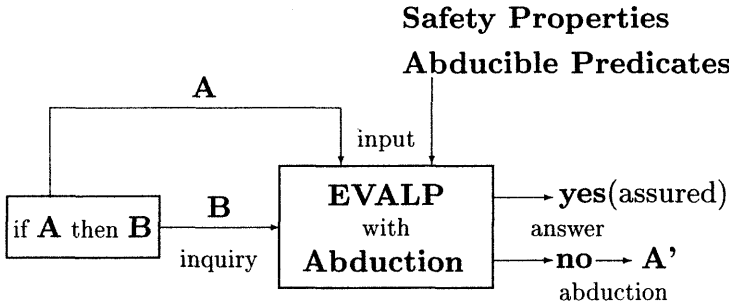


Figure 2: The Outline of the Verification System

3.2 The Safety Properties in EVALP

The basic terminologies of railway interlocking, which are represented in Geographic Data Language (GDL for short), and the safety properties for railway interlocking are translated into EVALP clauses.

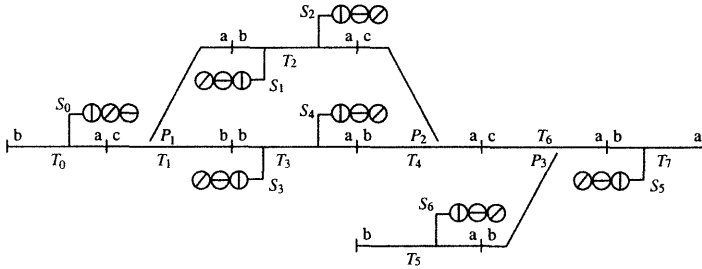


Figure 3: Signaling Schema for WEST

First of all, the GDL for signaling schema (Figure 3) from [6] is reviewed. The physical entities declared in the network are :

- track sections $\mathcal{T} = \{T_0, T_1, \dots, T_7\}$,
- points $\mathcal{P} = \{P_1, P_2, P_3\}$,
- signals $\mathcal{S} = \{S_0, S_1, \dots, S_6\}$,

and the logical control entities are :

- routes $\mathcal{R} = \{R_{02}, R_{04}, R_1, R_2, R_3, R_4, R_5, R_{51}, R_{53}, R_6\}$,
- sub-routes $\mathcal{U} = \{T_0^{ab}, T_0^{ba}, T_1^{ca}, \dots, T_7^{ba}\}$.

For example, the sub-route T_0^{ab} denotes the railway from a to b in the track section T_0 , the route R_{02} denotes the route from the signal S_0 to the signal S_2 , and the route R_{02} consists of the sub-routes T_0^{ba} , T_1^{ca} and T_2^{ba} . Each entity has logical or physical states.

Sub-route has two states *locked*(1) and *free*(f).

“The sub-route is locked” means that the sub-route is supposed to be occupied by a train and “free” means unlocked. eg. $T_0^{ba} 1$ denotes the sub-route T_0^{ba} is scheduled to be occupied by a train.

Route has two states *set*(s) and *unset*(xs).

“The route is set” means that all the sub-routes in the route are locked and “unset” means not set.

Track section has two states *occupied*(o) and *cleared*(c).

“The track section is occupied” means that the train is passing through the track section and “cleared” means that the train has already passed through the track section.

Point has four states :

controlled normal(cn) eg. P_1 cn denotes the point P_1 is controlled normal direction (ca or ac directions in the track section T_1) ;

controlled reverse(cr) eg. P_1 cr denotes the point P_1 is controlled reverse direction (cb or bc directions in the track section T_1) ;

controlled normal or free to move(cnf) eg. P_1 cnf denotes the point P_1 is controlled normal, or if it is not, P_1 can be moved to reverse position if the normal sub-routes are free ; and

controlled reverse or free to move(crf) eg. P_1 crf denotes the point P_1 is controlled reverse, or if it is not, P_1 can be moved to normal position if the reverse sub-routes are free.

The set \mathcal{Q}_{PRR} of PRRs is declared : $\mathcal{Q}_{PRR} = \{Q02, Q04, \dots, Q6, \dots\}$.

For example,

$Q02$ if P_1 crf, $T_1^{ac} f$, $T_2^{ab} f$ then $R_{02} s$, P_1 cr, $T_1^{ca} 1$, $T_2^{ba} 1$

is a PRR for the route R_{02} from the signal S_0 to the signal S_2 . The set \mathcal{Q}_{SRR} of SRRs is also declared : $\mathcal{Q}_{SRR} = \{SR02, SR04, \dots, SR6, \dots\}$.

For example,

$SR02$ if $R_{02} xs$, $T_1 c$, $T_2 c$ then $T_1^{ca} f$, $T_2^{ba} f$

is an SRR for the route R_{02} .

The safety properties consist of the following three conditions :

MX it is never the case that two or more of the sub-routes over a given track section are simultaneously locked ;

RT whenever a route is set, all its component sub-routes are locked ;

PT whenever a sub-route over a track section containing points is locked, the points are controlled in alignment with that sub-route.

The safety properties **MX**, **RT** and **PT** are represented in EVALP, then, the symbols $\{1, f, s, xs, cn, cnf, cr, crf, o, c\}$ that represent the states of entities are used in extended vector annotations instead of usual vector annotations (2-dimensional vectors). The following mappings between extended vector annotations are defined as epistemic negations : for each $\mu \in \{\alpha, \beta, \gamma\}$,

$$\begin{aligned}\neg_1([1, \mu]) &= [f, \mu], & \neg_1([f, \mu]) &= [1, \mu], & \neg_1([s, \mu]) &= [xs, \mu], & \neg_1([xs, \mu]) &= [s, \mu], \\ \neg_1([cn, \mu]) &= [cr, \mu], & \neg_1([cr, \mu]) &= [cn, \mu], & \neg_1([cnf, \mu]) &= [crf, \mu], \\ \neg_1([crf, \mu]) &= [cnf, \mu], & \neg_1([o, \mu]) &= [c, \mu], & \neg_1([c, \mu]) &= [o, \mu].\end{aligned}$$

For example, an EVALP clause $T(0, ab): [f, \alpha] \rightarrow T(0, ba): [f, \gamma]$ is intuitively interpreted as “if it is a fact that the sub-route T_0^{ab} is free, then the sub-route T_0^{ba} is permitted to be locked”.

The three safety property **MX**, **RT** and **PT** are translated into EVALP clauses.

[MX] Generally, the safety property **MX** represents that it is forbidden that two or more of the sub-routes over a given track section are simultaneously locked. The condition $\mathbf{MX}[T_0^{ab}, T_0^{ba}]$ can be interpreted as “if one of the sub-routes T_0^{ab}, T_0^{ba} is free, the other sub-route is permitted to be locked”, which is translated into EVALP clauses

$$T(0, ab): [f, \alpha] \rightarrow T(0, ba): [f, \gamma], \quad (11)$$

$$T(0, ba): [f, \alpha] \rightarrow T(0, ab): [f, \gamma]. \quad (12)$$

Similarly, the conditions $\mathbf{MX}[T_2^{ab}, T_2^{ba}]$, $\mathbf{MX}[T_3^{ab}, T_3^{ba}]$, $\mathbf{MX}[T_5^{ab}, T_5^{ba}]$ and $\mathbf{MX}[T_7^{ab}, T_7^{ba}]$ are also translated into EVALP clauses

$$T(2, ab): [f, \alpha] \rightarrow T(2, ba): [f, \gamma], \quad (13)$$

$$T(2, ba): [f, \alpha] \rightarrow T(2, ab): [f, \gamma], \quad (14)$$

$$T(3, ab): [f, \alpha] \rightarrow T(3, ba): [f, \gamma], \quad (15)$$

$$T(3, ba): [f, \alpha] \rightarrow T(3, ab): [f, \gamma], \quad (16)$$

$$T(5, ab): [f, \alpha] \rightarrow T(5, ba): [f, \gamma], \quad (17)$$

$$T(5, ba): [f, \alpha] \rightarrow T(5, ab): [f, \gamma], \quad (18)$$

$$T(7, ab): [f, \alpha] \rightarrow T(7, ba): [f, \gamma], \quad (19)$$

$$T(7, ba): [f, \alpha] \rightarrow T(7, ab): [f, \gamma]. \quad (20)$$

The track section T_1 contains the point P_1 , and the condition $\mathbf{MX}[T_1^{ac}, T_1^{ca}, T_1^{bc}, T_1^{cb}]$ can be interpreted as “if one of the normal(resp. reverse) side sub-routes T_1^{bc}, T_1^{cb} (T_1^{ac}, T_1^{ca}) is free and the point P_1 is permitted to be controlled normal(resp. reverse), the rest of the normal(resp. reverse) side sub-routes is permitted to be locked” so that the safety property **PT** can be considered. Therefore, the condition is translated into EVALP clauses

$$T(1, cb): [f, \alpha] \wedge P(1): [cr, \gamma] \rightarrow T(1, bc): [f, \gamma], \quad (21)$$

$$T(1, bc) : [\mathbf{f}, \alpha] \wedge P(1) : [\mathbf{cr}, \gamma] \rightarrow T(1, cb) : [\mathbf{f}, \gamma], \quad (22)$$

$$T(1, ca) : [\mathbf{f}, \alpha] \wedge P(1) : [\mathbf{cn}, \gamma] \rightarrow T(1, ac) : [\mathbf{f}, \gamma], \quad (23)$$

$$T(1, ac) : [\mathbf{f}, \alpha] \wedge P(1) : [\mathbf{cn}, \gamma] \rightarrow T(1, ca) : [\mathbf{f}, \gamma]. \quad (24)$$

Similarly, the conditions $\mathbf{MX}[T_4^{ac}, T_4^{ca}, T_4^{ab}, T_4^{ba}]$ and $\mathbf{MX}[T_6^{ac}, T_6^{ca}, T_6^{ab}, T_6^{ba}]$ are also translated into EVALP clauses

$$T(4, ba) : [\mathbf{f}, \alpha] \wedge P(2) : [\mathbf{cr}, \gamma] \rightarrow T(4, ab) : [\mathbf{f}, \gamma], \quad (25)$$

$$T(4, ab) : [\mathbf{f}, \alpha] \wedge P(2) : [\mathbf{cr}, \gamma] \rightarrow T(4, ba) : [\mathbf{f}, \gamma], \quad (26)$$

$$T(4, ca) : [\mathbf{f}, \alpha] \wedge P(2) : [\mathbf{cn}, \gamma] \rightarrow T(4, ac) : [\mathbf{f}, \gamma], \quad (27)$$

$$T(4, ac) : [\mathbf{f}, \alpha] \wedge P(2) : [\mathbf{cn}, \gamma] \rightarrow T(4, ca) : [\mathbf{f}, \gamma], \quad (28)$$

$$T(6, ca) : [\mathbf{f}, \alpha] \wedge P(3) : [\mathbf{cr}, \gamma] \rightarrow T(6, ac) : [\mathbf{f}, \gamma], \quad (29)$$

$$T(6, ac) : [\mathbf{f}, \alpha] \wedge P(3) : [\mathbf{cr}, \gamma] \rightarrow T(6, ca) : [\mathbf{f}, \gamma], \quad (30)$$

$$T(6, ba) : [\mathbf{f}, \alpha] \wedge P(3) : [\mathbf{cn}, \gamma] \rightarrow T(6, ab) : [\mathbf{f}, \gamma], \quad (31)$$

$$T(6, ab) : [\mathbf{f}, \alpha] \wedge P(3) : [\mathbf{cn}, \gamma] \rightarrow T(6, ba) : [\mathbf{f}, \gamma]. \quad (32)$$

[RT] The safety property **RT** represents that if all the sub-routes contained in one route are permitted to be locked, the route is permitted to be set. The condition $\mathbf{RT}(R_{02}, [T_1^{ca}, T_2^{ba}])$ can be interpreted as “if both the sub-routes T_1^{ca} and T_2^{ba} are permitted to be locked, the route R_{02} is permitted to be set”, which is translated into EVALP clause

$$T(1, ca) : [\mathbf{f}, \gamma] \wedge T(2, ba) : [\mathbf{f}, \gamma] \rightarrow R(02) : [\mathbf{xs}, \gamma]. \quad (33)$$

Although there are many routes in the network, only the routes R_1, R_3, R_{51}, R_{53} are translated into EVALP clauses as examples. The conditions $\mathbf{RT}(R_1, [T_1^{ac}, T_0^{ab}])$, $\mathbf{RT}(R_3, [T_1^{bc}, T_0^{ab}])$, $\mathbf{RT}(R_{51}, [T_6^{ac}, T_4^{ab}, T_2^{ab}])$ and $\mathbf{RT}(R_{53}, [T_6^{ac}, T_4^{ab}, T_3^{ab}])$ are translated into EVALP clauses

$$T(1, ac) : [\mathbf{f}, \gamma] \wedge T(0, ab) : [\mathbf{f}, \gamma] \rightarrow R(1) : [\mathbf{xs}, \gamma], \quad (34)$$

$$T(1, bc) : [\mathbf{f}, \gamma] \wedge T(0, ab) : [\mathbf{f}, \gamma] \rightarrow R(3) : [\mathbf{xs}, \gamma], \quad (35)$$

$$T(6, ac) : [\mathbf{f}, \gamma] \wedge T(4, ac) : [\mathbf{f}, \gamma] \wedge T(2, ab) : [\mathbf{f}, \gamma] \rightarrow R(51) : [\mathbf{xs}, \gamma], \quad (36)$$

$$T(6, ac) : [\mathbf{f}, \gamma] \wedge T(4, ab) : [\mathbf{f}, \gamma] \wedge T(3, ab) : [\mathbf{f}, \gamma] \rightarrow R(53) : [\mathbf{xs}, \gamma]. \quad (37)$$

[PT] The safety property **PT** represents the relation between point control and sub-route interlocking. The conditions $\mathbf{PTcn}(P_1, [T_1^{bc}, T_1^{cb}])$ and $\mathbf{PTcr}(P_1, [T_1^{ac}, T_1^{ca}])$ can be interpreted as “if one of the normal(resp. reverse) side sub-routes T_1^{bc}, T_1^{cb} (T_1^{ac}, T_1^{ca}) is free and P_1 is controlled normal(resp. reverse) or free to move, then the point P_1 is permitted to be controlled normal(resp. reverse)”, which are translated into EVALP clauses

$$T(1, bc) : [\mathbf{f}, \alpha] \wedge P(1) : [\mathbf{cnf}, \alpha] \rightarrow P(1) : [\mathbf{cr}, \gamma], \quad (38)$$

$$T(1, cb) : [\mathbf{f}, \alpha] \wedge P(1) : [\mathbf{cnf}, \alpha] \rightarrow P(1) : [\mathbf{cr}, \gamma], \quad (39)$$

$$T(1, ac) : [\mathbf{f}, \alpha] \wedge P(1) : [\mathbf{crf}, \alpha] \rightarrow P(1) : [\mathbf{cn}, \gamma], \quad (40)$$

$$T(1, ca) : [\mathbf{f}, \alpha] \wedge P(1) : [\mathbf{crf}, \alpha] \rightarrow P(1) : [\mathbf{cn}, \gamma]. \quad (41)$$

The conditions $\mathbf{PTcn}(P_2, [T_4^{ab}, T_4^{ba}])$, $\mathbf{PTcr}(P_2, [T_4^{ac}, T_4^{ca}])$, $\mathbf{PTcn}(P_3, [T_6^{ac}, T_6^{ca}])$ and $\mathbf{PTcr}(P_3, [T_6^{ab}, T_6^{ba}])$ for the points P_2 and P_3 are also translated into EVALP clauses

$$T(4, ab) : [\mathbf{f}, \alpha] \wedge P(2) : [\mathbf{cnf}, \alpha] \rightarrow P(2) : [\mathbf{cr}, \gamma], \quad (42)$$

$$T(4, ba) : [\mathbf{f}, \alpha] \wedge P(2) : [\mathbf{cnf}, \alpha] \rightarrow P(2) : [\mathbf{cr}, \gamma], \quad (43)$$

$$T(4, ac) : [\mathbf{f}, \alpha] \wedge P(2) : [\mathbf{crf}, \alpha] \rightarrow P(2) : [\mathbf{cn}, \gamma], \quad (44)$$

$$T(4, ca) : [\mathbf{f}, \alpha] \wedge P(2) : [\mathbf{crf}, \alpha] \rightarrow P(2) : [\mathbf{cn}, \gamma], \quad (45)$$

$$T(6, ac) : [\mathbf{f}, \alpha] \wedge P(3) : [\mathbf{cnf}, \alpha] \rightarrow P(3) : [\mathbf{cr}, \gamma], \quad (46)$$

$$T(6, ca) : [\mathbf{f}, \alpha] \wedge P(3) : [\mathbf{cnf}, \alpha] \rightarrow P(3) : [\mathbf{cr}, \gamma], \quad (47)$$

$$T(6, ab) : [\mathbf{f}, \alpha] \wedge P(3) : [\mathbf{crf}, \alpha] \rightarrow P(3) : [\mathbf{cn}, \gamma], \quad (48)$$

$$T(6, ba) : [\mathbf{f}, \alpha] \wedge P(3) : [\mathbf{crf}, \alpha] \rightarrow P(3) : [\mathbf{cn}, \gamma], \quad (49)$$

Now we consider Sub-Route Release in terms of the route unset conditions for routes. Sub-Route Release is issued sequentially along to the route to be unset. For example, suppose that the sub-route T_0^{ab} is being released. If the sub-routes T_1^{ac} or T_1^{bc} are permitted to be free and the track section T_0 is cleared, then T_0^{ab} is permitted to be free. Moreover, suppose that some different routes have sub-routes in common. If all the routes must be unset and the track section is cleared, then the common sub-routes is permitted to be free, or if all the precedent sub-routes are permitted to be free and the track section is cleared, then the sub-route is permitted to be free. For example, if both the routes R_{53} and R_{51} are unset, and the track section T_6 is cleared, the sub-route T_6^{ac} is permitted to be free. Although we need some more EVALP clauses to represent such sub-route release information, we omit those for space restriction.

3.3 Safety Verification Examples

A safety verification example for PRR based on abductive EVALP is provided. Generally, a PRR Qx has a form of Qx if A_1, \dots, A_m then $B_1, \dots, B_n \setminus$. PPRs can be checked their safety by consulting the safety properties as follows :

- (I) let an EVALP EP be the set $\{ (11), \dots, (49) \}$ of EVALP clauses representing the safety properties ;
- (II) translate each $A_i (1 \leq i \leq m)$ into EVALP clauses, and add all the EVALP clauses to the EVALP EP ; and
- (III) translate each $B_j (1 \leq j \leq n)$ into EVALP clauses, and inquire it from the EVALP EP obtained at (II), then if **yes** is returned, the safety for the PRR is assured, and if **no** is returned, compute the correct if-part by abduction with EVALP.

Example 3 We take a wrong PRR Q02' as an example of the safety verification :

$$Q02' \quad \text{if} \quad P_2 \text{ crf}, T_1^{ac} \mathbf{f}, T_2^{ab} \mathbf{f} \quad \text{then} \quad R_{02} \mathbf{s}, P_1 \text{ cr}, T_1^{ca} \mathbf{1}, T_2^{ba} \mathbf{1} \setminus.$$

We suppose $\{ P, T, R \}$ as the set of abducible predicates. The if-part of the PRR Q02' is translated into EVALP clauses :

$$P(2) : [\mathbf{crf}, \alpha], \quad (50)$$

$$T(1, ac) : [\mathbf{f}, \alpha], \quad (51)$$

$$T(2, ab) : [\mathbf{f}, \alpha], \quad (52)$$

which are added to the EVALP EP . Then, the then-part of the PRR Q02' is inquired to the EVALP EP and if **no** is returned, the correct if-part is abduced : the EVALP clause

$$T(2, ba) : [\mathbf{f}, \gamma] \quad (53)$$

is verified by the EVALP clauses (13) and (52) ; however, the EVALP clause

$$P(1):[\text{cn}, \gamma] \quad (54)$$

cannot be verified and **no** is returned ; then, by the abductive procedure SLDNFA ϵ , the EVALP clause

$$P(1):[\text{crf}, \alpha], \quad (55)$$

is abducted from the EVALP clauses (40) and (51) ; if the wrong EVALP clause (50) can be corrected to the EVALP clause (55), the PPR Q02' is verified.

4 Conclusions and Future Work

In this paper, we developed a safety verification system for railway interlocking mainly in station yards based on EVALP and its abduction. We are developing a safety verification system for ordinary train service based on the same verification method. If we have delay of train service, schedule changing of the train service and real time safety verification for the updated train service may be required. In order to realize such a real time safety verification system based on EVALP, EVALP has to be extended so as to be able to deal with interval time based reasoning, and hardware implementation of the system may be required. We are interested in hardware implementation of EVALP on a microchip, though we do not address it at all in this paper.

References

- [1] Blair, H.A. and Subrahmanian, V.S. : Paraconsistent Logic Programming. Theoretical Computer Science **68** (1989) 135–154
- [2] da Costa, N.C.A., Subrahmanian, V.S., and Vago, C. : The Paraconsistent Logics P7. Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **37** (1989) 139–148
- [3] Denecker, M. and De Schreye, D. : SLDNF: An Abductive Procedure for Abductive Logic Programs. J. Logic Programming, **34** (1998) 111–167
- [4] Gordon, M.J.C. and Melham, T.F., *Introduction to HOL*, Cambridge Univ. Press, (1993)
- [5] Lloyd, J.W. : *Foundations of Logic Programming* 2nd edition. Springer-Verlag, 1987
- [6] Morley, J.M. : Safety Assurance in Interlocking Design. Ph.D Thesis, University of Edinburgh, 1996
- [7] Nakamatsu, K. and Suzuki, A. : Annotated Semantics for Default Reasoning. Dai, R. (ed.) Proc. 3rd Pacific Rim International Conference on Artificial Intelligence, International Academic Publishers (1994) 180–186
- [8] Nakamatsu, K. and Suzuki, A. : A Nonmonotonic ATMS Based on Annotated Logic Programs. Agents and Multi-Agents Systems, LNAI **1441** Springer-Verlag (1998) 79–93
- [9] Nakamatsu, K., Abe, J.M. and Suzuki, A. : A Defeasible Deontic Reasoning System Based on Annotated Logic Programming. Proc. the Fourth International Conference on Computing Anticipatory Systems, American Institute of Physics, AIP Conference Proceedings 573, (2001) 609–620
- [10] Nakamatsu, K., Abe, J.M. and Suzuki, A. : Annotated Semantics for Defeasible Deontic Reasoning. Proc. the Second International Conference on Rough Sets and Current Trends in Computing, LNAI **2005**, Springer-Verlag, (2001) 432–440
- [11] Nakamatsu, K., Nagashima, J., Abe, J.M. and Suzuki, A. : An Automated Safety Verification System for Railway Interlocking Based on Extended Vector Annotated Logic Programming. Proc. 6th World Multi-conference on Systemics, Cybernetics and Informatics, Vol. XIV, (2001) 367–372

Complete Algorithm to realize CI model-based Control and Monitoring Strategies on Microcontroller Systems

Klaus-Dietrich Kramer, Steffen Patzwahl, Thomas Nacke*

Hochschule Harz - University of Applied Sciences

Department of Automation and Computer Science

Friedrichstraße 57-59, D-38855 Wernigerode

Phone: ++49(0)3943 / 659-317, Fax: ++49(0)3943 / 659-107

email: kkramer@hs-harz.de, spatzwahl@hs-harz.de

** Institute for Bioprocessing and Analytical Measurement Techniques e.V.*

Rosenhof, D-37308 Heiligenstadt

Phone: ++49(0)3606 / 671-163, Fax: ++49(0)3606 / 671-200

email: thomas.nacke@iba-heiligenstadt.de

Abstract. This paper describes a complete algorithm for the realization of model-based control and monitoring strategies adopting computational intelligence (CI) strategies to microcontroller systems. Process interfaces, data mining and realization of process models and controller structures are subjected to detailed discussion. A further key issue is exporting the complete algorithm to target hardware systems with microcontrollers and/or digital signal processors. The paper outlines a software tool for user-friendly implementation of the complete algorithm. The application presented is a fuzzy control (FC) system to control substrate supply to biogas pilot plants.

1. Introduction

In the past few years computational intelligence (CI) procedures – besides classical methods or to complement such – have gained increasing significance in modelling applications. Fuzzy control (FC) systems or applications of artificial neural networks are the state-of-the-art. CI methods have become especially indispensable in finding solutions in bioprocess engineering-related process classification, modelling or control, due to grossly non-linear interrelationships of process variables often in connection with no or hardly existing process expertise. Some applications are described in [5],[7],[11],[13]-[15],[18] and [19] below. Presently many tasks employing CI methods, with the exception of large-scale process control systems, remain restricted to PC-based solutions. Direct practical applications, especially in low- and medium-cost control tasks similarly to PID control modules in classical control engineering, could not yet be achieved with CI strategies. It is this status that the works presented in this paper go on from. A firmware development tool for embedded systems with microcontrollers (μC) is proposed that

permits besides model building and simulation also simple porting of the entire control task to a target platform (embedded system).

2. Complete algorithm

The complete algorithm (figure 1) is based on a profound process analysis ([2], [22]) that is aimed at evaluating the series of statistical values of *on-line* process data, if available in combination with relevant process expertise. Central objective is the preparation of a suitable process model. The initially essential aspect is the type of modelling. Depending on the target (model with an analog output quantity or classifier) there is a quite a number of model building options. Classical modelling methods use, e.g., differential equations or statistic classifiers, while CI-based models employ artificial neural networks and/or fuzzy logic. Frequently it is especially a combination of classical and CI-based models that yields the desired solution for a process model or a control strategy.

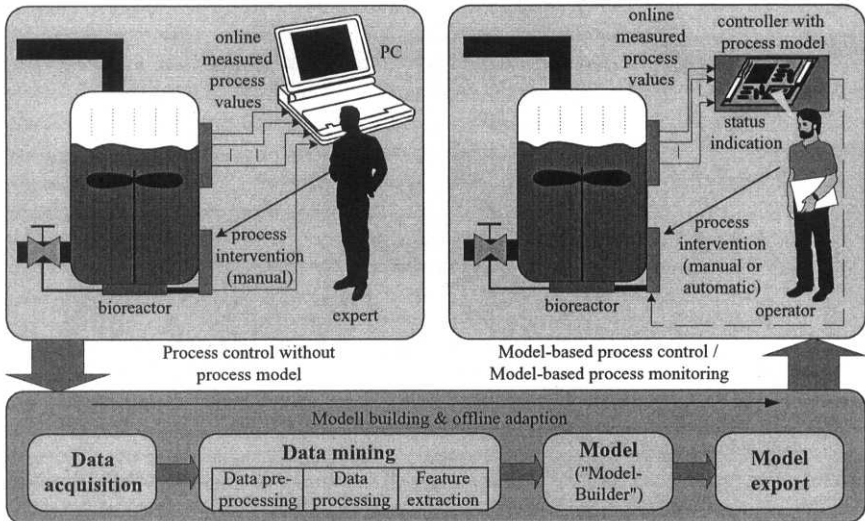


Figure 1: Complete algorithm for data-based development of models / controller structures

The tool „Model Builder“, a sub-application of the software system to realize the complete algorithm, was especially developed for model forming. The Model Builder integrates several modelling methods and makes them available to the user as a graphic development interface. Each model can be parameterized and put to data-based testing as required. Easy and swift model building and simulative verification enable the user to realise very quickly different modelling approaches to a given task. In particular the easy comparability of the developed models considerably facilitates finding an optimal modelling type. Also the implementation of controller structures is taken into consideration. A second sub-application of the software system is the tool „Application Builder“. This software tool is tasked with integrating the prepared model into the above-described complete algorithm of the design process. Its essential feature is data mining. In many cases, also especially in bioprocesses, optimal model input variables can only be identified via a preceding feature formation. This is why also simple methods for feature extraction from time data series have been implemented into this software tool. Data mining is complemented by data preprocessing and processing functions. Moreover, in addition, the

Application Builder parameterizes the *on-chip* periphery of the target hardware, especially analog-to-digital conversion (ADC), digital output (DOUT) and timer.

Finally the complete algorithm is exported into a run-time (RT) C code. The firmware development tool for embedded systems generates an RT-C code that contains all aspects of the complete algorithm in the form of programme modules. Also the model building strategies are integrated into the C code. Following compilation (C compiler for target hardware) the integrated and complete solution can be directly exported to the target hardware system. Thus, the μ C system with the respective firmware can be applied directly in the process independently of a PC. This μ C system has essential advantages compared with PC-based solutions in terms of ruggedness and price efficiency.

3. Software tool to realize the complete algorithm

The software tool to realize the complete algorithm is a software tool for MS-WindowsTM operating systems. It is divided into the sub-tools „Model Builder“ and „Application Builder“ and enables realization of the complete algorithm. Figure 2 shows the structure of this software system. All sub-functions are displayed for the user block-oriented via a graphic user interface.

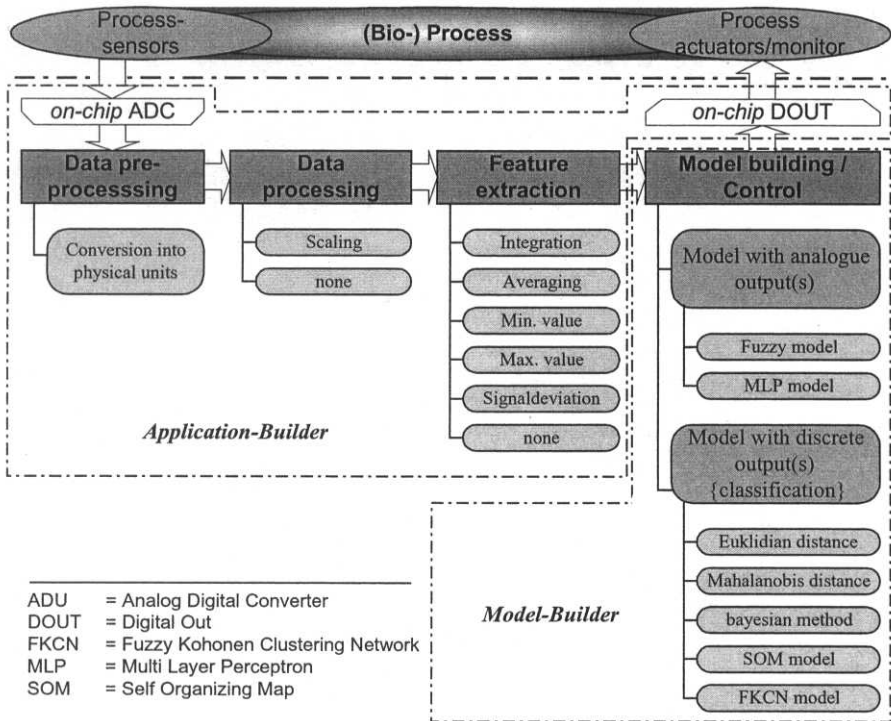


Figure 2: Structure of software tool to realize the complete algorithm

3.1. Data Mining

Below is a more detailed explanation of the data mining functions. The first data mining step is the preparation of data. Under consideration of its practical applicability the Application Builder concept assumes that direct sensor output signals from the plant are

used as input signals for the model to be developed. Typically, such sensor output signals are voltage values representing process signal values. A conversion rule for each input variable can be deposited in the module „data preprocessing“ which enables the calculation of physical units of process variables proceeding from sensor-provided voltage values.

The module „Data processing“ is available as second data mining step and can be used to scale input signals. Typically, all input signals are adjusted to the 0 - 1 range by means of Formula 1 below [24].

$$m_{nc}^* = \frac{m_{nc} - m_{nmin}}{m_{nmax} - m_{nmin}} \quad \forall c = 1, \dots, C, \quad \forall n = 1, \dots, N \quad (1)$$

The last step of data mining is provided in the Application Builder by the module „Feature extraction“. Feature extraction is aimed at the forming of features related to series of statistical input values. For one series of statistical values with 144 measured values (scanning rate: 10 min, observation period: 24 h), e.g., the arithmetic mean can be input as a feature describing a series of statistical values. The targets of feature extraction are data reduction and above all filtering of significant signal features to facilitate or enable later modelling. In view of the high significance of feature extraction for the monitoring or control algorithm the Application Builder is equipped with different methods for feature extraction from data. On account of later porting to the target hardware only simple methods have been employed. In practice it was found, however, that many applications can be particularly well realized with such simple feature extraction methods. Below is a detailed explanation of the implemented formula to calculate features using time data series [24][21].

Arithmetical mean

$$\bar{x} = \frac{1}{n} \cdot \sum_{i=0}^{n-1} x_i \quad (2)$$

Integral value (trapezoidal rule)

$$x_{\text{integral}} = \sum_{i=1}^{n-1} \frac{x_i + x_{i+1}}{2} \quad (3)$$

Minimum value

$$x_{\min} = \arg \min_{i=1}^n \{x\} \quad (4)$$

Maximum value

$$x_{\max} = \arg \max_{i=1}^n \{x\} \quad (5)$$

Range of variation

$$x_{\text{var.range}} = x_{\max} - x_{\min} \quad (6)$$

The Application Builder is used for channel-specific and block-oriented data mining parameterization. In order to ensure openness of the software, signals in the blocks „Data processing“ und „Feature extraction“ can also be used without algorithm. This is necessary, e.g., in the realization of structures where process model or process controller are directly using the process data.

3.2. Model building / Controller design

Below is a presentation of modules for model building and controller design (Model Builder).

Fuzzy Model

A fuzzy-model ([8],[23]) can be used for model building or process control in cases where the process to be modelled cannot be described mathematically by means of differential equations, but where experts can provide a qualitative or verbal description of the relations between process influencing variables. The Model Builder provides a complete development environment for fuzzy models and fuzzy controllers.

MLP Model

If a mathematical process description is not possible and in the absence of process expertise, artificial neural networks are one opportunity to still obtain a process model. However, precondition for application of artificial neural networks is the existence of process-describing data records. The Model Builder includes a Multi-Layer Perceptron (MLP) with a backprop learning algorithm ([17], [24]) which enables it to “learn” relations between inputs and outputs of a training data record file and apply such to new, not learned input data in a recall phase. MLP can be used as a process controller [11].

Statistical classifiers (Euclid, Mahalanobis, Bayes)

Statistical classifiers sort certain value pairs of model inputs (objects) on the basis of distance and/or truth dimensions into one or more predefined classes (model output) ([3],[8],[12],[16]). This classification is one form of model building. The classification results make process monitoring or automatic process intervention possible. The Model Builder includes distance and similarity dimensions after Euclid, Mahalanobis and Bayes. ([3],[12],[16],[24])

As required by the target of the classification method and according to the properties of input data records the user can define different classifiers and verify their results with test data records.

SOM Model

Self-organising artificial neural networks (self-organising maps = SOM) are capable of automatically recognising neighbourhood properties in input data patterns. Hence, SOM are ideally suited for data classification. Provided that there is a suitable network algorithm and a matching weight matrix allocation, a certain coherent neighbourhood area for each input data pattern is activated in the output configuration while the other properties remain inactive. Unsupervised learning in the training phase results in self-organization of weights as a function of frequency distribution of the provided input patterns and their neighbourhood properties ([9],[17]). In order to be usable as a classifier each neuron has to be assigned a meaning in a labelling phase. The SOM algorithm in the Model Builder combines the training and labelling phases into one step and, in a further step, can apply the trained SOM to new data.

FKCN Model

The Fuzzy Kohonen Clustering Network (FKCN) [1] is a neuro-fuzzy model where fuzzy logic and artificial neural networks ceased to exist independently side by side but blend into each other in the form of an overall model [24]. Especially self-organising artificial neural networks [9] and the fuzzy c-means algorithm ([1],[6],[24]) were combined. The main objective of the symbiosis of fuzzy logic and artificial neural networks is to combine the advantages and compensate the shortcomings of the individual methods.

All above-mentioned model-building methods can also be used for process controller development. To this end the process model has to be designed by using an actuating variable as model output.

3.3. Data output (on-chip DOUT)

Monitoring or control algorithm findings in the form of model output variable or controller output variable have to be output within the scope of the direct process concept via the *on-chip* periphery of the target hardware. To this end, the model output variable is time-dependent output via a digital output bit in analog-output models or with controller algorithms. The turnon time represents the value of the analog variable. Thus, process actuators, such as pumps, can be selected. In discrete-output models (classificators) the classification result is output via a digital output port in the target hardware and can fed, e.g., also into a monitoring system.

3.4. Code generation and target hardware

Finally, the complete algorithm is translated into a C code as per the relevant user specifications. The generated C code contains also programme blocks for *on-chip* periphery control of the target hardware, mainly ADC, digital-port output and timer functions. Moreover, an interrupt system is initiated and used in the algorithm. The C code is optimized for the target hardware and, due to the special characteristics of every target hardware system, can only run on the indicated system after compilation with a suitable C compiler. Presently the Application Builder supports two target hardware systems, a system with μC 80C167 (Figure 3) and a system with TriCore™ (both comp. Infineon) [10].

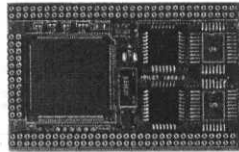


Figure 3: Minimodule® made by Comp. phytec with μC 80C167

Summary of the main performance characteristics of the target hardware systems:

Infineon 80C167

- Processing range: 16 bit, clock frequency (CPU): 20 MHz
- 16 bit ALU (arithmetic and logic unit)
- 10-Bit-ADC, usable for 16 channels per multiplex operation
- Two 16-bit timer/counter units
- Parallel input and output ports (total of 111 lines)
- Internal clock generator
- Two serial interfaces

Infineon-TriCore™

The TriCore™ combines the benefits of digital signal processors and microcontrollers. Thus, the periphery is served by means of process interfaces and highly efficient mathematical function for signal functions are provided.

The TriCore™ is an RISC processor with the below outstanding parameters [10]:

- Processing range: 32 bit, clock frequency (CPU): 40 MHz
- Bit-reverse address modes for DSP algorithms
- *On-chip* SRAM for data and time-critical code
- Two ADC with 8-, 10- or 12-bit resolution and 16 analog inputs each
- Eleven 16-bit digital I/O ports, two analog ports
- Internal clock generator

- Two serial interfaces, two CAN interfaces

Chart 1 shows an exemplary RT C code. The individual steps of the complete algorithm are reflected in the forms of the program modules.

```

/* ADC */
//...
ADCIC=0x66;
ADCON=0xF280;
A_IN[0]=(ADDA&0x3ff);
A_IN[0]=(input[0]*5)/0x3ff;
//...
/* Data mining (Data preprocessing) */
DM_IN_1[0]=(A_IN[0]*2)-29.877;
//...
/* Data mining (Data processing) */
DM_IN_2[0]=(DM_IN_1[0] - 2.3) / 25.45;
//...
/* Data mining (feature extraction) */
DM_IN_3[0]+=DM_IN_2[0]; // Total
if (actual_sample==144)
{
    DM_IN_3[0]=DM_IN_3[0]/144 // mean
    DM_IN_3[0]=0;
}
//...
/* Model (process monitor with SOM) */
//...
if (net<min)
{
    min=net;
    maxinp[i]=0;
}
//...
SOMOut[i]=exp(-min*min);
for (j=0; j<TrainOutCol+1; j++)
{
    if (ValueCounter[j]==compare)
        MR_OUT=j;
}
//...
/* DOUT */
P2 = MR_OUT;

```

Chart 1: Exemplary RT C code of the complete algorithm

4. Application in biogas plants (applicative example)

4.1. Biogas generation and control engineering problems

Biogas generation from organic residues is a complex process, especially in anaerobic co-fermentation, and proceeds via a number of intermediate products. The substrate composition is not constant and is subjected to substance-related and seasonal variations. Economic use of biogas plants for energy generation requires a gas yield as continually as possible at an optimal carbon degradation. Meeting this requirement makes it necessary to control the quantity of fed substrate as a function of the desired gas yield volume. Conventional control systems meet their technical limits when having to control such anaerobic fermentation processes for biogas generation. The difficulty are non-linear relations between substrate supply and biogas extraction which are linked to large time constants. In addition, those control engineering problems in anaerobic fermentation are aggravated by order-related co-fermentation of different organic wastes. The latter entails differing space loads of the fermentation plant. The anaerobic fermentation process is relatively complex and very difficult to describe in mathematical and model terms due to the non-linearity between substrate feeding, biomass growth and gas yield [15] [20].

4.2. Fuzzy model to control substrate feeding

A process-description database was generated as basis to build this model. Figure 4 shows the lab-scale biogas plant (fermenter volume 6 l) with extensive *online* measured data acquisition equipment.

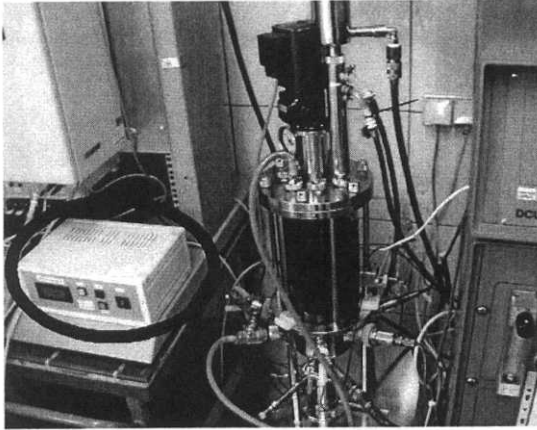


Figure 4: Picture of the lab-scale biogas pilot plant (iba – Heiligenstadt)

Following process and analysis in conjunction with correlation analyses the below measured data were assessed to be process- and model-relevant.

- Biogas volume (BGV) [dm^3]
- CH_4 concentration in the generated gas (CH_4) [%]
- CO_2 concentration in the generated gas (CO_2) [%]
- O_2 concentration in the generated gas (O_2) [%]
- pH (pH) [-]

The fuzzy model [15][20] developed by means of the Model Builder as part of the overall plant concept is shown in Figure 5.

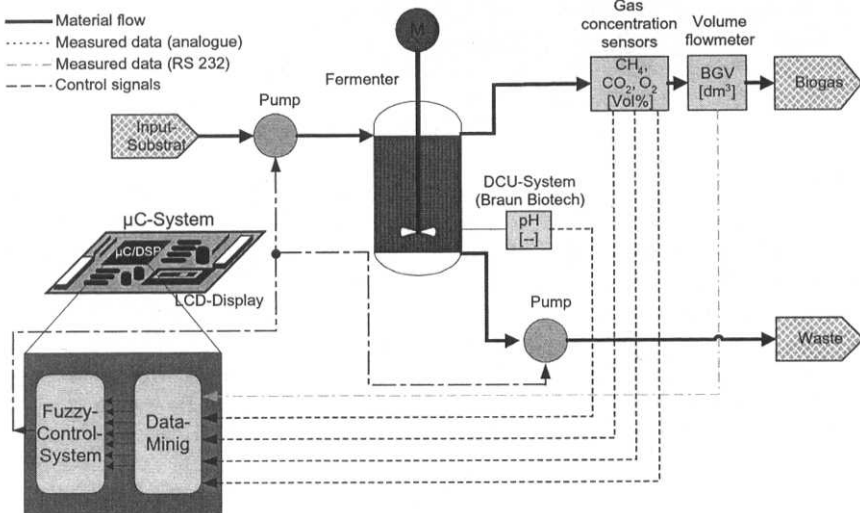


Figure 5: Structure of the fuzzy model with preceding feature extraction

This FC model works with seven input variables, generated by feature extraction during data mining:

- Gas quantity per day (GM) [dm^3/d]
- CO_4 content per day (CH4_d) [%]
- CO_2 content per day (CO2_d) [%]
- O_2 content per day (O2_d) [%]
- Quotient of CH_4 and CO_2 content per day (CH4/CO2_d) [-]
- Variation of gas volume between current day and preceding day (dGM) [dm^3/d]
- Arithmetic mean of pH (pH_m) [-]

The optimal additional dosing rate (Q_{in}) is determined as an output variable and can be directly used for process control as an actuating variable. The control basis (22 Controlling) and the parameters of the affiliation functions of the linguistic terms were prepared together with process experts (Fuzzy expert system).

The model relates to the substrates fats and hydrolysis as input materials into the fermentation process. The principal purpose of the fuzzy model is to ensure process safety, i.e. a process failure is prevented. Moreover, the model optimizes total gas and CH_4 yields [15][20].

4.3. Tests performed at the biogas pilot plant

As a field test the compact unit with μC 80C167 (Figure 4) was tested for 1.5 months in the biogas lab-scale pilot plant where the substrate feeding and discharge pumps were directly selected via the system. Plant operator interventions during the trials were restricted to planned system interferences. All *on-line* measured variables were parallelly recorded by the PC-supported measuring system for monitoring purposes. The dosing data calculated and displayed by the compact unit, which in real terms formed the basis of substrate feeding via pump control, were also manually recorded. The result curves of the trial run are shown in Figure 6.

The evaluation shows that the process was optimally controlled by the compact unit, apparent from:

- a relatively constant and high CH_4 concentration in the produced biogas
- a gas volume $> 4 \text{ l/d}$
- a pH within the optimum process range
- a low coefficient from CH_4 and CO_2 concentration as an indication of process safety

In addition, the system was disturbed twice in order to test its reaction to interference. First, the pump was switched off on 13.10.2001 so that – contrary to the calculated dosing rate – a dosing was not made. The fuzzy system reacted to this disturbance, under consideration of the retention time, by increasing the dosing rate on 15.10.2001.

As a second interference, on 21.10.2001 the reactor was fed with the base substrate hydrolysis and oil. Oil hampers the process of anaerobic fermentation and endangers the process sequence. The fuzzy system reacted consistently, under consideration of the retention time, by decreasing the dosing rate on 23.10.2001.

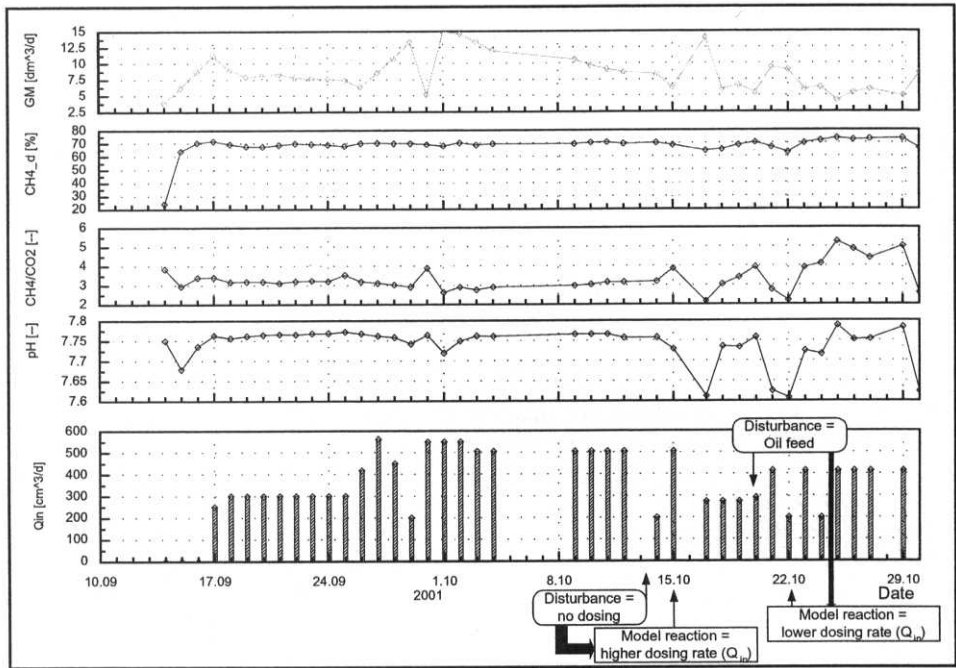


Figure 6: Simulation results of the FC system (progression of features and actuating variables)

5. Summary

This paper presents a complete algorithm for the realization of model-based control and monitoring strategies in microcontroller systems based on computational intelligence (CI) strategies. It discusses all relevant sub-aspects from process signals via data mining to process model and control algorithm with signal output. A further key issue is the export of the monitoring or control algorithm, developed on a PC by using the described software tools and under consideration of the design steps, into a Run-time (RT)C code for the target hardware. The RT C code formed as a result of the complete algorithm constitutes an application of its own in the target hardware system with μ C or DSP which can be employed for process monitoring or plant control independently of a PC.

The works referenced in this paper received public funding within the scope of a research project of the Federation of industrial research associations „Otto von Guericke“ e.V. (aif) – Project owner of the BMBF (Federal Ministry of Education and Research) under the programme aFuE (Public grant Ref. N°: 17.113.01).

References

- [1] **Bezdek, J., Pal, S. K.:** *Fuzzy Models for Pattern Recognition*. New York: IEEE Press, 1992.
- [2] **Bocklisch, S. F.:** *Prozessanalyse mit unscharfen Verfahren*. Berlin: Verlag Technik, 1987.
- [3] **Canty, M. J.:** *Fernerkennung mit neuronalen Netzen – Theorie – Algorithmen – Programme*. Renningen-Malmsheim: Expert-Verlag, 1999, S. 122-125.
- [4] **Günther, M.:** *Kontinuierliche und zeitdiskrete Regelungen*. Stuttgart: B.G. Teubner-Verlag, 1997.
- [5] **Estaben, M., Polit, M., Steyer, J. P.:** *Fuzzy Control for an anaerobic digester*. Control Eng. Practice, Vol. 5, No. 98, p.1303-1310, 1997.
- [6] **Höppner, F., Klawonn, F., Kruse R.:** *Fuzzy-Clusteranalyse, Verfahren für die Bilderkennung, Klassifizierung und Datenanalyse*. Braunschweig, Wiesbaden: Vieweg-Verlag, 1997.
- [7] **Isermann, R.:** *Zur Anwendung der Fuzzy-Logik in der Regelungstechnik*. Automatisierungstechnische Praxis 38, München, Wien: Oldenbourg Verlag, S. 24-26, 1996.
- [8] **Koch, M., Kuhn, T., Wernstedt, J.:** *Fuzzy Control – Optimale Nachbildung und Entwurf optimaler Entscheidungen*. München, Wien: Oldenbourg-Verlag, 1996.
- [9] **Kohonen, T.:** *Self-organizing and associative memory*. Heidelberg, Berlin, New York: Springer Verlag, 1989.
- [10] **Kramer, K.-D.:** *TriCore – eine 32-Bit-MC-DSP-Architektur*. HTW Mittweida: Tagungsband 5. Workshop „Microcontroller-Applikationen“, S. 9-16, 2000.
- [11] **Linko, S., Zhu, Y.-H., Linko, P.:** *Applying neural networks as software sensors for enzyme engineering*. TIBTECH, Vol. 17, p. 155-162, 1999.
- [12] **Mahalanobis, P. C.:** *On the Generalized Distance on Statistic*. Proc. Nat. Inst. India, No. 2, p. 49-55, 1936.
- [13] **Patzwahl, S., Nacke, T., Kramer, K.-D.:** *Selforganizing neural networks to supervise the content of water in biological filtration plants*. 7th European Congress on Intelligent Techniques and Soft Computing, Proceedings of EUFIT '99 Aachen, Germany, Final programme, Abstracts of the Papers & Proceedings on CD-ROM, Aachen, 1. Edition Aachen: Wissenschaftsverlag, S. 104 ff., 1999.
- [14] **Patzwahl, S., Nacke, T., Kramer, K.-D.:** *SOMonCON - ein windowsbasiertes Softwarewerkzeug zur Implementierung Selbstorganisierender Neuronaler Netze auf Mikrocontrollersystemen*. Arbeitsgemeinschaft Fuzzy-Logik und Softcomputing Norddeutschland (AFN), AFN-Berichte 2000, Clausthal Zellerfeld: Papierflieger-Verlag, 2000
- [15] **Patzwahl, S. et al.:** *Microcontroller-Based Fuzzy System to Optimize the Anaerobic Digestion in Biogas Reactors*. Computational Intelligence, International Conference, 7th Fuzzy Days, Dortmund, Germany, October 1-3, 2001, Proceedings, London, Heidelberg, Berlin: Springer-Verlag, p. 2-10, 2001.
- [16] **Ripley, B. D.:** *Pattern Recognition and Neural Networks*. Cambridge: Cambridge University Press, 1992.
- [17] **Schmitter, E.-D.:** *Neuronale Netze - Einführung, Programmierbeispiele, praktische Anwendungen*. 1. Auflage, Holzkirchen: W. Hofacker Verlag, 1991.
- [18] **Souza De, M. B., Ferreira, L. S., Folly, R. O. M.:** *Development of an alcohol fermentation control system based on biosensor measurements interpreted by neural networks*. Sensors and Actuators B 3713, p.1-6, 2001.
- [19] **Stenz, R., Kuhn, J.:** *Vergleich: Fuzzy-Automatisierung und konventionelle Automatisierung einer Batch-Destillationskolonne*. Automatisierungstechnische Praxis 35, München, Wien: Oldenbourg Verlag, S. 228-295, 1993.
- [20] **Tautz, T.:** *Entwicklung und praktische Realisierung eines mikrocontroller-basierten CI-Regelungssystems für Biogasanlagen*. Diplomarbeit, Hochschule Harz, Wernigerode, 2001.
- [21] **Wang, X. Z.:** *Data mining and knowledge discovery for process monitoring and control*. London, Heidelberg, Berlin: Springer-Verlag, 1999.
- [22] **Wernstedt, J.:** *Experimentelle Prozessanalyse*. Berlin: Verlag Technik, 1989.
- [23] **Zadeh, L.:** *Fuzzy Sets*. Information and Control, 8, S.338-353, 1965.
- [24] **Zimmermann, H.-J.:** *Datenanalyse – Anwendung von DataEngine mit Fuzzy-Technologien und Neuronalen Netzen*. Düsseldorf: VDI-Verlag, 1995.

Using Genetic Algorithms for Minimizing the Production Costs of Hollow Core Slabs

Vanessa C. Castilho¹; Maria C. Nicoletti²; Mounir K. El Debs¹

¹DEE – EESC – USP – Brasil {castilho,mkdebs}@sc.usp.br

²DC – UFSCar – Brasil {carmo@dc.ufscar.br}

Abstract. Genetic algorithms (GAs) are adaptive methods based on the genetic process of biological organisms that have been successfully applied to a variety of tasks, in areas such as function optimization, parameter tuning, learning, etc. The main goal of this work is to investigate a set of selection strategies which are used by a typical GA for minimizing the cost function of prefabricated hollow core slabs and to demonstrate GA's robustness over a conventional method with respect to the complexity of the problem and quality of the solution.

1. Introduction

Much of the research in the area of Optimization has been aimed at the identification and evaluation of the most promising and efficient methods for solving problems. In general, for representing an optimization problem there is initially the need of identifying the variables involved in the problem, their scope, as well as constant values, so to equate the solving of the problem to the solving of its mathematical formulation. The solution of the problem consists, basically, in finding a solution (the best) which identifies a point of maximum or minimum of an objective function, subjected to some restrictions.

A large proportion of research work and experiments that have been carried out in the area of Structural Engineering focuses on the minimization of the cost of structures and employs mainly conventional methods of optimization. As examples of this tendency there is the work of Prakash, Agarwala and Singh [18], which minimizes the cost of reinforced concrete sections using the Lagrangian and simplex methods, the work of Chakrabarty [3], which approaches the minimization of the costs of reinforced concrete beams using a geometrical programming model that gives the unique least-cost design of a beam, considering the cost of materials and shuttering and the structural requirements, the work of Cohn and Lounis [4] which describes the minimization of total structural cost, concerning the optimal design of structural concrete bridge systems, using a Lagrangian Method and the work of Koskito and Ellingwood [15], that uses the structural reliability theory to minimize the life-cycle cost of prefabricated concrete elements and structures.

In spite of the relative success of conventional methods when solving optimization problems in relation to structural analysis, such methods have some limitations. These limitations include identifying optimal global solutions, dealing with real and discrete variables and dealing with a multi-objective optimization problem. These methods make

assumptions concerning continuity and the existence of gradient information and cannot be used in certain problems of structural optimization, when the objective function is not differential. For these reasons research work in this area has aimed at the identification of alternative methods which can be more flexible and can give the same or even improve the results obtained with conventional techniques. Genetic algorithm (GA) is one of these alternative options. A GA is an adaptive general-purpose search algorithm which has successfully been applied to various Structural Engineering problems dealing with optimization (see for instance, [5], [11] and [12]).

This paper presents an investigation of applying a GA to search for the optimal solution of the problem of minimizing the production costs of hollow core slabs. It focuses on a set of *ad-hoc* selection strategies, which gives rise to a family of GA variations; the experiments described in this paper tried to identify the most suitable variation for finding the solution to the problem described here. For comparison, the paper also presents the results obtained using the roulette wheel selection operator as well as those obtained using a conventional optimization method, namely the Augmented Lagrangian, implemented as the system EASY [13,14] (available for download at www.ime.unicamp/~martinez).

The remainder of this work is organized as follows. Section 2 describes a few basic concepts of GA which are relevant to the work and presents the six GA variations' pseudo-code used in this work. Section 3 describes the problem i.e., the costs involved in the production of prefabricated hollow core slabs and describes how these costs have been combined and represented as a function with constraints. Section 4 describes the experiments for optimizing the cost function, using the six GA variations, the Augmented Lagrangian and a typical GA implementing the roulette wheel selection operator, and discusses the results. Finally, in Conclusions, the next steps for continuing this work are pointed out.

2. The GA Variations Family

Genetic algorithms are adaptative methods based on the genetic process of biological organisms that have been successfully applied to a variety of tasks, in areas such as function optimization, parameter tuning, learning, etc. GAs can deal with complex problems in a computationally simple and yet powerful way and are not limited by a few assumptions in the way that conventional optimization methods are. The basic principles of GA have been rigorously established by Holland in [10] and can be found in many references (see for instance [1], [2] and [17]).

In GA the term population is used for naming a set of potential solutions to the problem; each individual solution is called a chromosome. Generally the initial population is initialized with a pre-defined number of chromosomes which are randomly created; usually the number of individuals per population remains constant during the whole process. Inspired by the biological natural selection process, the GA through selection operator chooses the chromosomes from the current population in order to determine which individual candidates will be part of the 'reproduction' process. In this work the basic selection operator of a GA has been turned into a selection strategy; it implements a selective process as well as a restoration process. In order to reflect the considered selection strategies, the typical GA described in Figure 1 has been modified giving rise to six GA variations.

The GA variations described in Figures 2 to 7 have in common the fact that the selection process selects those individuals which necessarily will undergo crossover. Taking into consideration the fact that only a percentage of individuals undergo crossover, a restoring mechanism which restores the population to its original size is necessary.

```

procedure GA
begin
   $t \leftarrow 0$ 
  initialise( $p(t)$ )
  evaluate( $p(t)$ )
  while (not termination-condition)
  do
    begin
       $t \leftarrow t + 1$ 
      select  $p(t)$  from  $p(t-1)$ 
      crossover( $p(t)$ )
      mutation( $p(t)$ )
      evaluate( $p(t)$ )
    end
  end

```

Figure 1. Typical Genetic Algorithm

Procedures *modified_GA_1* and *modified_GA_2* differ in how they implement the restoring mechanism. In both procedures all the generated offspring are part of the new population. In procedures *modified_GA_3* and *modified_GA_4* the set of generated offspring compete with all the individuals of the current population in order to be able to be selected for the next population. The set of generated offspring is extended to *pop_size* elements by adding individuals from the current population. The new population is extracted from the union of the current population with this extended set.

Procedures *modified_GA_5* and *modified_GA_6* implement what is called *steady-state replacement*, where in each generation only a few (typically two) individuals are replaced. As commented in [1], "in the steady-state case, we not only have to consider how to select two individuals to be parents, but we also have to select two unlucky individuals from the population to be killed off, to make way for the offspring. Several schemes are possible, including: 1. selection of parents according to fitness and selection of replacements at random; 2. selection of parents at random and selection of replacements by inverse fitness and 3. selection of both parents and replacements according to fitness/inverse fitness". Both procedures only differ in how the parents are selected. Some auxiliary procedures used in the description of the modified GAs are left undefined; their meaning can be easily inferred from their names.

<pre> procedure <i>modified_GA_1</i> {<i>pop_size</i>: size of the population <i>p_c</i>: probability of crossover <i>r</i>: $p_c \times \text{pop_size}$} begin $t \leftarrow 0$ <i>initialise</i> ($p(t)$) <i>evaluate</i> ($p(t)$) while (not <i>termination-condition</i>) do begin $t \leftarrow t + 1$ <i>select_parents</i>($p(t-1)$,<i>r</i>,<i>Parents</i>) <i>crossover</i>(<i>Parents</i>,<i>Offsprings</i>) </pre>	<pre> procedure <i>select_parents</i>(<i>Pop</i>,<i>N</i>,<i>P</i>) {<i>Pop</i>: population <i>N</i>: integer such that $N \leq \text{Pop}$ <i>P</i>: the <i>N</i> fittest individuals from <i>Pop</i>} begin <i>sort_decreasing_fitness</i>(<i>Pop</i>,<i>Sorted</i>) <i>select</i>(<i>Sorted</i>,<i>N</i>,<i>P</i>) end procedure <i>select</i>(<i>A</i>,<i>B</i>,<i>C</i>) {<i>C</i> is the set of the <i>B</i> first individuals from <i>A</i>} procedure <i>random_choice</i>(<i>A</i>,<i>B</i>,<i>C</i>) </pre>
---	---

<pre> random_choice(p(t-1),pop_size - r,Rest) p(t) ← mutation(Offsprings ∪ Rest) evaluate(p(t)) end end </pre>	<pre> {C is the set of B randomly chosen individuals from A} </pre>
--	---

Figure 2. Procedure *modified_GA_1*

<pre> procedure modified_GA_2 {pop_size: size of the population pc: probability of crossover r: $p_c \times \text{pop_size}$} begin t ← 0 initialise (p(t)) evaluate (p(t)) while (not termination-condition) do begin t ← t + 1 select_parents(p(t-1),r,Parents,Rest) crossover(Parents,Offsprings) p(t) ← mutation(Offsprings ∪ Rest) evaluate(p(t)) end end end </pre>	<pre> procedure select_parents(Pop,N,P,R) {Pop: population N: integer such that $N \leq \text{Pop}$ P: the N fittest individuals from Pop R: the remaining $\text{Pop} - N$ individuals from Pop} begin sort_decreasing_fitness(Pop,Sorted) select(Sorted,N,P,R) end procedure select(A,B,C,D) {transfers the first B elements from A to C and the remaining elements to D} </pre>
--	---

Figure 3. Procedure *modified_GA_2*

<pre> procedure modified_GA_3 {pop_size: size of the population pc: probability of crossover r: $p_c \times \text{pop_size}$} begin t ← 0 initialise (p(t)) evaluate (p(t)) while (not termination-condition) do begin t ← t + 1 select_parents(p(t-1),r,Parents) crossover(Parents,Offsprings) random_choice(p(t-1),pop_size-r,R) Aux ← mutation(Offspring ∪ R) sort_decreasing_fitness(p(t-1) ∪ Aux,Sorted) select(Sorted,pop_size,p(t)) evaluate(p(t)) end end end </pre>	<pre> procedure select_parents(Pop,N,P) {Pop: population N: integer such that $N \leq \text{Pop}$ P: the N fittest individuals from Pop} begin sort_decreasing_fitness(Pop,Sorted) select(Sorted,N,P) end procedure select(A,B,C) {C is the set of the B first individuals from A} procedure random_choice(A,B,C) {C is the set of B randomly chosen individuals from A} </pre>
---	---

Figure 4. Procedure *modified_GA_3*

<pre> procedure modified_GA_4 {pop_size: size of the population pc: probability of crossover r: $p_c \times \text{pop_size}$} begin t ← 0 initialise (p(t)) evaluate (p(t)) while (not termination-condition) do begin </pre>	<pre> procedure select_parents(Pop,N,P,R) {Pop: population N: integer such that $N \leq \text{Pop}$ P: the N fittest individuals from Pop R: the remaining $\text{Pop} - N$ individuals from Pop} begin sort_decreasing_fitness(Pop,Sorted) select(Sorted,N,P,R) end </pre>
--	--

<pre> t ← t + 1 select_parents(p(t-1),r,Parents,R) crossover(Parents,Offsprings) Aux ← mutation(Offspring ∪ R) sort_decreasing_fitness(p(t-1) ∪ Aux,Sorted) select(Sorted,pop_size,p(t)) evaluate(p(t)) end end </pre>	<pre> procedure select(A,B,C) {C is the set of the B first individuals from A} </pre>
--	--

Figure 5. Procedure *modified_GA_4*

<pre> procedure modified_GA_5 {pop_size: size of the population number of parents = r=2} begin t ← 0 initialise(p(t)) evaluate(p(t)) while (not termination-condition) do begin t ← t + 1 select_parents(p(t-1),r,Parents,Worst_Candidates) crossover(Parents,Offsprings) Set_of_2r ← Worst_Candidates ∪ Offsprings sort_decreasing_fitness(Set_of_2r,New_set_of_2r) select(New_set_of_2r,r,Best_r,Worst_r) Aux ← p(t-1) - Worst_Candidates Pop ← Aux ∪ Best_r p(t) ← mutation(Pop) evaluate(p(t)) end end end </pre>	<pre> procedure select_parents(Pop,N,P,U) {Pop: population N: integer such that $N \leq Pop$ P: the N fittest individuals from Pop} begin sort_decreasing_fitness(Pop,Sorted) select(Sorted,N,P) select(inv(Sorted),N,U) end procedure select(A,B,C,D) {transfers the first B elements from A to C and the remaining elements to D} procedure inv(L) {inverts the order of L} </pre>
--	--

Figure 6. Procedure *modified_GA_5*

<pre> procedure modified_GA_6 {pop_size: size of the population number of parents = r=2} begin t ← 0 initialise(p(t)) evaluate(p(t)) while (not termination-condition) do begin t ← t + 1 select_parents(p(t-1),r,Parents,Worst_Candidates) crossover(Parents,Offsprings) Set_of_2r ← Worst_Candidates ∪ Offsprings sort_decreasing_fitness(Set_of_2r,New_set_of_2r) select(New_set_of_2r,r,Best_r,Worst_r) Aux ← p(t-1) - Worst_Candidates Pop ← Aux ∪ Best_r p(t) ← mutation(Pop) evaluate(p(t)) end end end </pre>	<pre> procedure select_parents(Pop,N,P,U) {Pop: population N: integer such that $N \leq Pop$ P: the N fittest individuals from Pop} begin sort_decreasing_fitness(Pop,Sorted) select_random(Sorted,N,P) select(inv(Sorted),N,U) end procedure inv(L) {inverts the order of L} </pre>
--	--

Figure 7. Procedure *modified_GA_6*

3. Problem Definition

The optimization problem consists of the minimization of a function which represents the total cost involved in the production of a hollow core slab. In addition to the issues defined by the problem, the definition of the function also takes into consideration recommendations stated by The Brazilian Norms NBR-7197 and NBR-2000 (still a developing project). The variables which define the total cost function are: x_1 – thickness of slab (cm), x_2 – amount of steel in cross section (cm^2) and x_3 – compressive strength of concrete (MPa). A cross-section of the slab is shown in Figure 8. The function used in the experiments was inspired by the cost function proposed in [15], without the cost associated with failure, which is inherent in the method the authors used.

The function representing the total cost to be minimized is given by equation (1) with relation to the production of the element, under the restrictions of the ultimate limit states (flexural strength and shear) and serviceability limit states (excessive deflection, flexural cracking) [7].

$$f(x) = 0.01Lx_1 + 0.0159Lx_2 + 0.00144Lx_1x_3 + 1.31Lx_1 / x_3 + 14.44Lx_2 / (x_3)^2 \quad (1)$$

where:

$f(x)$ – objective function

L – span of slab

x_1 – thickness of slab (cm)

x_2 – amount of steel in cross section (cm^2)

x_3 – compressive strength of concrete (MPa)

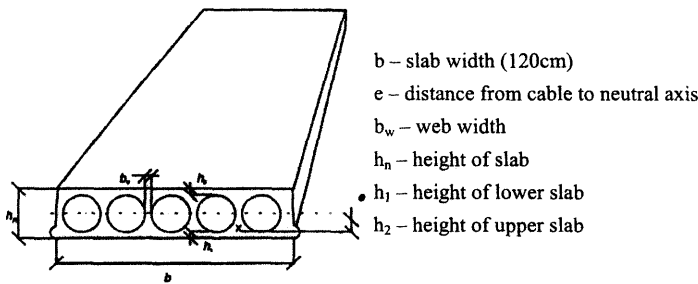


Figure 8. Hollow-core slab

Acceptable performance of concrete slabs is measured against the following limit states:

- serviceability limit states

a) decompression and formation of fissuration

– frequent combination

$$g_1(x_i) \leq 0 \quad \text{where} \quad g_1(x_i) = \sigma_{lg1} + \sigma_{lg2} + 0.3\sigma_{lq} + \sigma_{lp_{00}} - \sigma_{ct}$$

– quasi-permanent combination

$$g_2(x_i) \leq 0 \quad \text{where} \quad g_2(x_i) = \sigma_{lg1} + \sigma_{lg2} + 0.2\sigma_{lq} + \sigma_{lp00}$$

b) excessive deflection:

$$g_3(x_i) \geq 0 \quad \text{where} \quad g_3(x_i) = a_{lim} - a$$

• ultimate limit states (flexural strength and shear)

a) flexural tensile failure:

$$g_4(x_i) \leq 0 \quad \text{where} \quad g_4(x_i) = M_d - M_u$$

b) shear capacity:

$$g_5(x_i) \leq 0 \quad \text{where} \quad g_5(x_i) = V_d - V_u$$

where σ_{lg1} – stress due to dead weight of slab

σ_{lg2} – stress due to dead weight of *insitu* concrete

σ_{ct} – tension stress

σ_{lq} – stress due to live load

σ_{lp00} – stress due to prestressing

M_d – resisting moment

M_u – ultimate load moment

a_{lim} – limit deflection $(\frac{L}{300})$

a – deflection of slab under distributed load

V_d – shear load

V_u – shear capacity

Besides the previous restrictions, due to the problem characteristics, the range of values of the three variables were limited to the following intervals: $15 \leq x_1 \leq 40$; $2 \leq x_2 \leq 12$; $30 \leq x_3 \leq 50$. When applying GA to problems with restrictions, a commonly used technique is to add a penalty term to the objective function. Essentially this technique "transforms the constrained problem into an unconstrained problem by penalizing infeasible solutions, in which a penalty term is added to the objective function for any violation of the constraints" [8]. Generally the penalty function ($\text{pen}(x)$) is added to the objective function ($f(x)$) so to construct the fitness function ($F(x)$), as shown in equation (2). Usually the penalty function has value zero ($\text{pen}(x)=0$) when x is feasible and a positive value, otherwise.

$$F(x) = f(x) + \text{pen}(x) \quad (2)$$

There is not a general guidance for defining the penalty function for optimization problems. This work adopts one defined by the linear equation:

$$\text{pen}(x) = 1500 C$$

$$\text{where } C = \sum_{i=1}^m c_i$$

m : number of restrictions

c_i : value associated with the $g_i(x)$ restriction, determined as:

$$\begin{array}{ll} \text{if } g_i(x) \leq 0 & \text{if } g_i(x) \geq 0 \\ \text{then } c_i = 0 & \text{then } c_i = 0 \\ \text{else } c_i = 1 & \text{else } c_i = 1 \end{array}$$

4 Description of the Experiments and Analysis of the Results

The six modified GAs described in the Section 2 were implemented sharing the characteristics listed in Table 1.

Table 1. Main Characteristics of the GA Variations

characteristics	values
elitism	1 individual
size of the population	100
representation of the solution	real
crossover	arithmetical operator with $\lambda = 2/3$
probability of crossover	0.85
probability of mutation	0.01
stop criterion	1000 generations

Due to the characteristics of the problem, where the possible solutions tend to occur close to each other in the search space, the most suitable representation for potential solutions to the problem is the real representation, as suggested in [6],[9]. The GA variations' implementations deal with chromosomes that are encoded as vectors of three real numbers corresponding to the three variables involved in the function (1). The crossover operator implemented is the arithmetical operator and the best results were obtained using $\lambda = 2/3$. The adopted mutation is the random mutation where the value of a variable is substituted for another one, randomly chosen within the range of the variable.

Since AGs are very sensitive to the initial population, the data shown in Table 2 (and Figure 9) is for the average values obtained using five randomly selected initial populations. Figure 9 shows the cost function throughout a GA's run, for each of the six implemented strategies. It also shows the values obtained using three other strategies named *roulette_wheel_1*, *roulette_wheel_2* and *roulette_wheel*. The first two differ from their counterparts *modified_AG_1* and *modified_AG_2* only in the way the parents are selected from the current population; the roulette wheel operator is used for selecting the $p_c \times \text{pop_size}$ elements from the current population, which will undergo crossover. The third strategy, roulette wheel, implements a typical GA, using the roulette wheel as the selection operator, as described in [17]. The Augmented Lagrangian reached the same result in several executions of the program, each using a random set of initial points.

Focusing initially on the six proposed strategies, it can be observed that, in spite of their results differing slightly, the different approaches adopted for restoring the population to its original size have not had much affect on the final results. As can be seen in Table 2 (and corresponding Figure 9) the best solution is obtained with the strategy described by procedure *modified_AG_1*, where the population is randomly restored and the worst, with

procedure *modified AG 2*, where the population is restored using individuals from the set of those that were not selected for crossover. In spite of the differences among the results obtained by the six different GA variations, they all achieved better results than the three GAs implementing the roulette wheel selection operator.

Table 2. Cost function and its variables (average of five runs)

VALUE OF THE FUNCTION (US\$/m ²)					VARIABLES		
strategy	F(x)	standard deviation	absolute best	absolute worst	x ₁ (cm)	x ₂ (cm ²)	x ₃ (MPa)
modified GA 1	3.66	0.0105	3.65	3.67	20.11	2.72	40.1
modified GA 2	3.88	0.0639	3.76	3.92	21.66	2.72	37.6
modified GA 3	3.77	0.0886	3.69	3.91	21.27	2.58	39.1
modified GA 4	3.79	0.0917	3.73	3.95	21.42	2.60	40.0
modified GA 5	3.69	0.0653	3.64	3.79	20.40	2.67	41.7
modified GA 6	3.74	0.0479	3.68	3.80	20.15	2.87	44.7
roulette wheel 1	4.34	0.249	3.93	4.52	24.21	2.88	36.1
roulette wheel 2	4.44	0.511	3.93	5.29	24.17	3.26	44.1
roulette wheel	4.32	0.164	4.09	4.51	24.37	2.86	35.0
Augmented Lagrangian	3.75	—	—	—	20.64	2.76	38.0

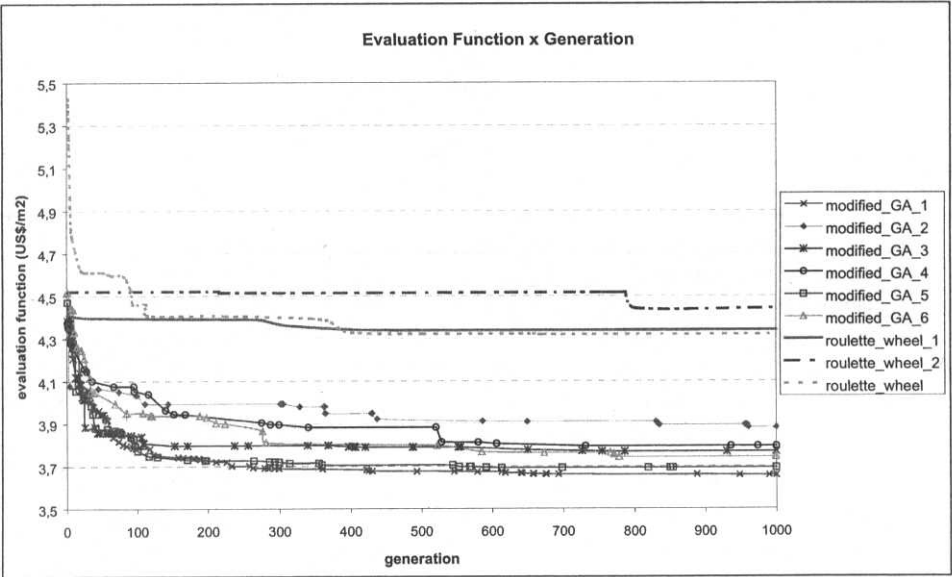


Figure 9. Cost function using different GA's variations

5. Conclusions

This work presents a family of *ad-hoc* GA's variations used for minimizing the cost function associated with the production of prefabricated hollow core slabs. For comparison it presents the results obtained using three variations of a GA using the roulette wheel

selection operator and also, presents the results obtained using the Augmented Lagrangian conventional method of optimization. In spite of one of the *ad-hoc* variations achieving the best results, it is necessary to experiment with other selection strategies to effectively elect the strategy implemented by *modified_GA_1* as the best one for the solution of this problem. It is important to mention that some of the strategies proposed have a higher computational cost (eg *modified_GA_3*). We intend next to investigate if the GA variations presented in this paper enjoy similar theoretical properties as the classical GA.

Acknowledgement

To CAPES for the doctoral scholarship granted to Vanessa Cristina de Castilho and Leonie C. Pearson for the insightful comments on this paper.

References

- [1] Beasley, D. *et al* (1993a). An Overview of Genetic Algorithms: Part 1, Fundamentals, University Computing, v.15, n.4, p.170-181.
- [2] Beasley, D. *et al* (1993b). An Overview of Genetic Algorithms: Part 2, Fundamentals, University Computing, v.15, n.2, p.58-69.
- [3] Chakrabarty, B. K. (1992). Models for Optimal Design of Reinforced Concrete Beams. *Computers & Structure*, v.42, n.3, p.447-451.
- [4] Cohn, M. Z. & Lounis, Z. (1994). Optimal Design of Structural Concrete Bridge Systems. *Journal of Structural Engineering*, ASCE, v.120, n.9, p.2653-2674, September.
- [5] Coello, C. C.; Hernández, F. S. & Farrera, F. A. (1997). Optimal Design of Reinforced Concrete Beams Using Genetic Algorithm. *Expert Systems with Applications*, v.12, n.1, p.101-108.
- [6] Davidor, Y. (1990). Genetic Algorithms and Robotics: a Heuristic Strategy for Optimization. World Scientific Publishing Co., Singapore.
- [7] El Debs, M. K. (2000). Precasted Concrete: Basics and Applications (in Portuguese), São Carlos. Projeto REENGE, EESC-USP.
- [8] Gen, M. & Cheng, R. (1997). Genetic Algorithms and Engineering Design. New York. John Wiley & Sons.
- [9] Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization and Machine Learning. U.S.A., Addison-Wesley Publishing Company.
- [10] Holland, J. H. (1975). Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.
- [11] Jenkins, W.M. (1997). On the Application of Natural Algorithms to Structural Design Optimization. *Engineering Structures*, v.19, n.4, p.302-308.
- [12] Leite, J.P.B. & Topping, B.H.V. (1998). Improved Genetic Operators for Structural Engineering Optimization. *Advances in Engineering Software*, v.29, n.7-9, p.529-562.
- [13] Martínez, J.M. (1997). Augmented Lagrangians and the Resolution of Packing Problems, Technical Report 08/97, Institute of Mathematics, University of Campinas, Brazil.
- [14] Martínez, J.M. (1998). A Two-phase Model Algorithm with Global Convergence for Nonlinear programming. *Journal of Optimization Theory and Applications* 96, p.397-436.
- [15] Koskisto, O. J. & Ellingwood, B. R. (1997). Reliability-based Optimization of Plant Precast Concrete Structures. *Journal of Structural Engineering*, ASCE, v.123, n.3, p.298-304, March.
- [16] Lounis, Z. & Cohn, M. Z. (1993). Optimization of Precast Prestressed Concrete Bridge Girder Systems. *PCI Journal*. v.123, n.3, p.60-77, July-August.
- [17] Michalewicz, Z. (1992). Genetic Algorithms + Data Structures = Evolution Programs. Berlin, Springer-Verlag.
- [18] Prakash, A.; Agarwala, S. K. & Singh, K. K. (1988). Optimum Design of Reinforced Concrete Sections. *Computers & Structure*, v.30, n.4, p.1009-1011.

Recognizing Malicious Intention in an Intrusion Detection Process

Frédéric Cuppens¹, Fabien Autrel¹, Alexandre Miège² & Salem Benferhat³

1: ONERA Toulouse, 2 Av. E. Belin, 31055 Toulouse Cedex, France,

2: ENST Paris, 46 rue Barrault, 75014 Paris CEDEX, France,

3: IRIT, 118 route de Narbonne, 31062 Toulouse CEDEX, France

email: {cuppens, autrel, miege}@cert.fr, benferhat@irit.fr

Abstract.

Generally, the intruder must perform several actions, organized in an *intrusion scenario*, to achieve his or her malicious objective. We argue that intrusion scenarios can be modelled as a planning process and we suggest modelling a malicious objective as an attempt to violate a given security requirement. Our proposal is then to extend the definition of attack correlation presented in [CM02] to correlate attacks with intrusion objectives. This notion is useful to decide if a sequence of correlated actions can lead to a security requirement violation. This approach provides the security administrator with a global view of what happens in the system. In particular, it controls unobserved actions through hypothesis generation, clusters repeated actions in a single scenario, recognizes intruders that are changing their intrusion objectives and is efficient to detect variations of an intrusion scenario. This approach can also be used to eliminate a category of false positives that correspond to false attacks, that is actions that are not further correlated to an intrusion objective.

1 Introduction

The main objective of computer security is to design and develop computer systems that conform to the specification of a security policy. A security policy is a set of rules that specify the authorizations, prohibitions and obligations of agents (including both users and applications) that can access to the computer system. An intruder might be viewed as a malicious agent that tries to violate the security policy.

Notice that sometimes the intruder might perform his intrusion by using a single action. However, more complex intrusions generally require several steps to be performed. For instance, there are two steps in the *Mitnick* attack. In the first step, the intruder floods a given host H . Then the intruder sends spoofed SYN messages corresponding to H address to establish a TCP connection with a given server S . When S sends a SYN-ACK message, H would normally send a RESET message to close the connection. But this is not possible since H is flooded. This enables the intruder to send an ACK message to illegally open a connection with S . Notice also that opening a TCP connection with S is probably not the intruder's final objective. It is likely that the intruder will then attempt to get an access on S for instance by performing a *rlogin*. This means that the *Mitnick* attack will actually represent preliminary steps of a more global intrusion. In the following, we shall call *intrusion scenario* the complete sequence of actions that enables the intruder to achieve his intrusion objective.

In this context, current intrusion detection technology only detects elementary attacks that correspond to the steps of a given intrusion scenario. They neither provide a global view of the intrusion scenarios in progress nor of the intrusion objectives the intruders attempt to achieve. Therefore, our goal in this paper is twofold. First, we suggest an approach to recognize various steps of an intrusion scenario. We shall call *attack correlation* this first functionality. It is actually an extension of the approach suggested in [CM02]. Second, when the attack correlation succeeds in gathering several actions, we want to decide whether the current observations actually correspond to malicious activities or not. We call *malicious intention recognition* this second functionality. Combining these two functionalities would enable the security administrator to have a global understanding of what happens in the system in order to prepare an adequate reaction. Notice that sometimes, this reaction might be launched before the intrusion scenario is completed, that is before the intrusion objective is actually achieved.

The remainder of this paper is organized as follow. Section 2 introduces preliminary definitions to fix the vocabulary. Section 3 presents our approach to modelling the intrusion process. Derived from planning process models in Artificial Intelligence, this model includes a representation of both attacks and intrusion objectives. Section 4 then presents our approach for modelling the intrusion *detection* process. Based on these materials and following [CM02], we define the notion of attack and alert correlation and also correlation between an attack and an intrusion objective. Section 5 further refines our approach by introducing hypothesis generation in the correlation process. This is useful to complete detection of an intrusion scenario when some steps in this scenario are not detected (false negatives). This is also useful to anticipate over the intruder intentions when several actions that match an intrusion scenario have been correlated. Section 6 illustrates our approach on several examples of intrusion scenarios. Finally section 7 concludes the paper.

2 Preliminary definitions

2.1 Intrusion objective and intrusion scenario

Intrusion objective (intrusion detection point of view) An intrusion objective is the final purpose of an intruder, which justifies all its actions. So, from its point of view, the intrusion objective is obvious. By contrast, from the intrusion detection point of view, it is more difficult to determine the possible intrusion objectives and to differentiate them from non malicious activities. As an intruder aims at committing a forbidden action, we suggest deriving the possible intrusion objectives from the security policy: *any security policy violation is a potential intrusion objective*. We give three examples corresponding to integrity, confidentiality, and availability violation: gaining a non authorized root access right (Objective 1), having a non authorised read access to a sensitive file (Objective 2), performing a denial of service attack on the DNS (Objective 3).

Intrusion scenario As an intrusion objective will often needs several actions to be reached, the intruder needs to draw up an *intrusion scenario*. It is an organised set of correlated actions, which have to be executed following a certain order. Let us present three intrusion scenarios corresponding to the intrusion objectives described just before.

1. **Illegal NFS mount:** the intruder, say *badguy*, wants to obtain a root access on a target. *badguy* can perform the following actions: (1) *rpcinfo* to know if *portmapper* is running

on the target, (2) with the *showmount* command, *badguy* sees the target exportable partitions, (3) *mount* enables *badguy* to mount one of this partition, say the root partition, (4) by modifying the *.rhost* file of this partition, (5) *badguy* gets a root access on the target system, (6) *rlogin* is the final step to access the target system with root privileges.

2. **DoS on the DNS:** this intrusion scenario leads to a DoS attack on the DNS server. A possible scenario is: (1) *nslookup* to locate the DNS server, (2) *ping* to check whether the DNS server is active, (3) *scan* port 139 (NetBios) to get evidence that Windows is active on the DNS, (4) *winnuke* that performs a DOS attack on Windows machine.
3. **Illegal file access:** we shall consider the following intrusion scenario example where an unauthorised user *bad_guy* tries to read *secret-file* [ZMB02]: (1) *bad_guy* creates a file (*touch file*), (2) *bad_guy* blocks the printer, by opening the paper trays, (3) *lpr -s* enables *bad_guy* to print *file*, (4) *bad_guy* deletes *file*, (5) *bad_guy* creates a symbolic link from *file* to *secret-file*: *ln -s file secret-file*, (6) *bad_guy* unblocks the printer and *secret-file* will be printed.

2.2 Malicious and suspicious actions

Malicious action A malicious action enables the intruder to directly achieve an intrusion objective. For instance, thanks to the *Winnuke* attack, an intruder can do a denial of service on a Windows server.

Action correlation (informal definition) $Action_1$ is correlated with $Action_2$ if $Action_1$ may enable the intruder to then perform $Action_2$.

Suspicious action A suspicious action is defined as an action that can be correlated to a malicious action or to another suspicious action.

According to this definition, a suspicious action may be an inoffensive action, or may also be a way to execute a malicious action on a following step. For example, scanning port 139 (NetBios) is not a dangerous action. But, if port 139 is open, the intruder knows that Windows is running and can perform the *Winnuke* attack.

Attack An attack is a malicious action or a suspicious action.

This is quite a weak definition of “attack” since it also includes suspicious actions. However, we guess it is close to the intrusion detection terminology since many alerts actually correspond to only suspicious actions.

2.3 Fusion and correlation process

Fusion process and fusion alert The simple alerts generated by different IDS detecting the same attack are merged into a single cluster. This is called *fusion process*. It determines first which are the merging criteria for each type of attack, and then, during the intrusion detection process, uses those criteria to constitute clusters. At last, it generates a *fusion alert* to inform all the security devices of the creation and the content of a new cluster. The fusion process is not the purpose of this paper; see [Cup01, VS01] for different proposals.

Correlation process and scenario alert The *correlation process* receives the fusion alerts and tries to correlate them one by one using correlation rules. When a complete or a partial intrusion scenario is detected, a *scenario alert* is generated. The purpose of this paper is to extend this correlation process suggested in [CM02].

3 Modelling intrusion from the intruder point of view

In our approach the intrusion process is modelled as a planning activity. We assume that the intruder wants to achieve intrusion objectives and, for this purpose, the intruder can use a set of attacks. The intruder's problem is to find a sequence of attacks that transforms a given initial state into a final state. The initial state corresponds to the system state when the intruder starts his intrusion. And the final state has the property that the intrusion objective is achieved in this state. We check this approach on several intrusion scenarios, including the three scenarios presented in section 2 but also other scenarios such as the *Mitnick* attack. Due to space limitation, we shall only illustrate scenario 3 "illegal file access" as a planning process. But before, we need to present our approach to model attacks and intrusion objectives.

3.1 Action representation

In the planning context, actions are represented by their pre and post conditions. Pre conditions correspond to conditions the system's state must satisfy to perform the action. Post conditions correspond to effects of executing the action on the system's state. Note that in the modelling of intrusion process system's state is only partially known.

In our model, an attack is similarly represented using three fields: its name, pre condition and post condition. Attack name is a functional expression that represents the name of the attack and its different parameters. Attack pre-condition specifies the *minimal* logical conditions to be satisfied for the attack to succeed and attack post-condition is a logical condition that specifies the *minimal* effect of the attack when this attack succeeds.

The pre-condition and post-condition of an attack correspond to conditions over the *system's state*. For this purpose, we use a language, denoted L_1 , which is based on the logic of predicates. Predicates are used to describe properties of the system state relevant to the description of an attack. In this language, we assume that terms starting by an upper case letter correspond to variables and other terms are constants.

The predicates are combined using the logical connectives "and" (conjunction denoted by a comma) and "not" (negation). Currently, we do not allow using disjunction in the pre and post conditions of an attack. This means that negation only applies to predicates, not to conjunctive expressions. Taking into account disjunctions is left for future work.

Figure 1 shows how various actions of the example *illegal file access* are represented in this model. To model this scenario, we actually specify 8 actions. The 6 first actions *touch*, *block*, *lpr-s*, *remove*, *ln-s* and *unblock* correspond to the various actions performed by the intruder in the *illegal file access* scenario as presented in section 2.

Our model includes two additional actions *print-process* and *get-file* that usually do not appear in this example. Action *print-process(Printer, Link)* models what happens on *Printer* when *Link* is queued: a file is printed if *Printer* is not blocked. This printed file will be *File* if there is a logical link between *Link* and *File*. Action *get-file(Agent, File)* corresponds to the physical action performed by *Agent* to get *File* after it is printed. This last action actually enables *Agent* to obtain a read access to *File*.

Action touch(<i>Agent</i>, <i>File</i>) Pre: <i>true</i> Post: <i>file(File)</i> , <i>owner(Agent, File)</i>	Action block(<i>Agent</i>, <i>Printer</i>) Pre: <i>printer(Printer)</i> , <i>physical_access(Agent, Printer)</i> Post: <i>blocked(Printer)</i>
Action lpr-s(<i>Agent</i>, <i>Printer</i>, <i>File</i>) Pre: <i>printer(Printer)</i> , <i>file(File)</i> , <i>authorized(Agent, read, File)</i> Post: <i>queued(File, Printer)</i>	Action remove(<i>Agent</i>, <i>File</i>) Pre: <i>owner(Agent, File)</i> Post: <i>not (file(File))</i>
Action ln-s(<i>Agent</i>, <i>Link</i>, <i>File</i>) Pre: <i>not (file(Link))</i> Post: <i>linked(Link, File)</i>	Action unblock(<i>Agent</i>, <i>Printer</i>) Pre: <i>printer(Printer)</i> , <i>blocked(Printer)</i> , <i>physical_access(Agent, Printer)</i> Post: <i>not (blocked(Printer))</i>
Action print-process(<i>Printer</i>, <i>Link</i>) Pre: <i>queued(Link, Printer)</i> , <i>linked(Link, File)</i> , <i>not (blocked(Printer))</i> Post: <i>printed(Printer, File)</i> , <i>not (queued(Link, Printer))</i>	Action get-file(<i>Agent</i>, <i>File</i>) Pre: <i>printed(Printer, File)</i> , <i>physical_access(Agent, Printer)</i> Post: <i>read_access(Agent, File)</i>

Figure 1: Modelling the illegal file access scenario

Intrusion_Objective illegal_root_access(<i>Host</i>) State_Condition: <i>root_access(Agent, Host)</i> , <i>not (authorized(Agent, root, Host))</i>
Intrusion_Objective illegal_file_access(<i>File</i>) State_Condition: <i>read_access(Agent, File)</i> , <i>not (authorized(Agent, read, File))</i>
Intrusion_Objective DOS_on_DNS(<i>Host</i>) State_Condition: <i>dns_server(Host)</i> , <i>dos(Host)</i>

Figure 2: Examples of intrusion objectives

3.2 Modelling intrusion objective

An intrusion objective is modelled by a system state condition that corresponds to a violation of the security policy. An intrusion objective contains two fields : its name and a set of conditions denoted by *State_condition*. Figure 2 provides three examples of intrusion objectives that respectively correspond to violation of the three requirements specified in the previous security policy example. For instance, intrusion objective *illegal_file_access(File)* is achieved if *Agent* has a read access to the file *File* but he is not authorized to read it.

3.3 Domain rules

Domain rules are used to represent general properties of system's state through possible relations between predicates. For convenience matters, these domain rules are also represented using a pre and post condition. However, the pre and post conditions of a domain rule are evaluated on the same system state: if the pre condition of a domain rule is true in a given state, then the post condition is also true in the *same* state.

Domain_rule <i>owner_right</i> (File) Pre: <i>owner</i> (Agent, File) Post: <i>authorized</i> (Agent, read, File), <i>authorized</i> (Agent, write, File)
Domain_rule <i>remove_right</i> (File) Pre: not <i>file</i> (File) Post: not (<i>owner</i> (Agent, File)), not (<i>authorized</i> (Agent, read, File)), not (<i>authorized</i> (Agent, write, File))

Figure 3: Examples of domain rules

Figure 3 provides two examples of domain rule. Rule *owner_right*(File) says that the Agent owner of a given File is automatically authorized to have read and write access to this file. Rule *remove_right*(File) says that if File does no longer exist, then there is no longer an owner for this file and read and write access to File are also removed to every Agent.

3.4 Planning intrusion scenario

Using the three previous sections, we can now show how the intrusion scenario *illegal file access* is modelled as a planning process. For this purpose, let us consider an intruder, say *bad_guy*, and a file containing sensitive data, say *secret_file*. Let us assume that *bad_guy* wants to achieve the intrusion objective *illegal_file_access*(*secret_file*). This means that *bad_guy* wants to achieve a final system state such that the following condition is satisfied:

- *read_access*(*bad_guy*, *secret_file*), not (*authorized*(*bad_guy*, read, *secret_file*))

Let us also assume that *bad_guy* starts in the following initial state:

- *file*(*secret_file*), not (*read_access*(*bad_guy*, *secret_file*)),
printer(*ppt*), *physical_access*(*bad_guy*, *ppt*)

That is, in the initial state, *secret_file* exists but *bad_guy* has not a read access to this file and there is a printer *ppt* and *bad_guy* has a physical access to this printer.

Now, the planning problem consists in finding a sequence of actions that transforms the initial state into the final state. Figure 4 presents a possible solution to this problem. It is easy to check that objective *illegal_file_access*(*secret_file*) is achieved in the state resulting from these 8 steps. Notice that there is another solution that corresponds to starting by blocking the printer and then creating *guy_file* using the *touch* command, the other steps being identical to the other solution.

According to our definitions presented in section 2, only *get_file*(*bad_guy*, *secret_file*) corresponding to step 8 is a malicious action since it enables the intruder to achieve the intrusion objective. Steps 1 to 7 are only suspicious actions in the sense that they enable the intruder to then perform step 8.

4 Attack and alert correlation

Our approach for intrusion scenario detection uses the same materials as the ones introduced in section 3. Based on these materials, we shall extend the definition of attack correlation suggested in [CM02] by defining the notion of objective correlation.

Step 1: <i>touch</i> (<i>bad_guy</i> , <i>guy_file</i>)
Step 2: <i>block</i> (<i>bad_guy</i> , <i>ppt</i>)
Step 3: <i>lpr-s</i> (<i>bad_guy</i> , <i>ppt</i> , <i>guy_file</i>)
Step 4: <i>remove</i> (<i>bad_guy</i> , <i>guy_file</i>)
Step 5: <i>ln-s</i> (<i>bad_guy</i> , <i>guy_file</i> , <i>secret_file</i>)
Step 6: <i>unblock</i> (<i>bad_guy</i> , <i>ppt</i>)
Step 7: <i>print-process</i> (<i>ppt</i> , <i>guy_file</i>)
Step 8: <i>get-file</i> (<i>bad_guy</i> , <i>secret_file</i>)

Figure 4: Planning the illegal file access scenario

Attack A	Attack B	Unifier
<i>touch</i> (<i>Agent</i> ₁ , <i>File</i> ₁)	<i>remove</i> (<i>Agent</i> ₂ , <i>File</i> ₂)	<i>Agent</i> ₁ = <i>Agent</i> ₂ , <i>File</i> ₁ = <i>File</i> ₂
<i>remove</i> (<i>Agent</i> ₁ , <i>File</i> ₁)	<i>ln-s</i> (<i>Agent</i> ₂ , <i>Link</i> ₂ , <i>File</i> ₂)	<i>File</i> ₁ = <i>Link</i> ₂
<i>block</i> (<i>Agent</i> ₁ , <i>Printer</i> ₁)	<i>unblock</i> (<i>Agent</i> ₂ , <i>Printer</i> ₂)	<i>Agent</i> ₁ = <i>Agent</i> ₂ , <i>Printer</i> ₁ = <i>Printer</i> ₂
<i>lpr-s</i> (<i>Agent</i> ₁ , <i>Printer</i> ₁ , <i>File</i> ₁)	<i>print-process</i> (<i>Printer</i> ₂ , <i>Link</i> ₂)	<i>File</i> ₁ = <i>Link</i> ₂ , <i>Printer</i> ₁ = <i>Printer</i> ₂
<i>ln-s</i> (<i>Agent</i> ₁ , <i>Link</i> ₁ , <i>File</i> ₁)	<i>print-process</i> (<i>Printer</i> ₂ , <i>Link</i> ₂)	<i>Link</i> ₁ = <i>Link</i> ₂ , <i>Printer</i> ₁ = <i>Printer</i> ₂
<i>unblock</i> (<i>Agent</i> ₁ , <i>Printer</i> ₁)	<i>print-process</i> (<i>Printer</i> ₂ , <i>Link</i> ₂)	<i>Printer</i> ₁ = <i>Printer</i> ₂
<i>print-process</i> (<i>Printer</i> ₁ , <i>Link</i> ₁)	<i>get-file</i> (<i>Agent</i> ₂ , <i>File</i> ₂)	<i>File</i> ₁ = <i>File</i> ₂ , <i>Printer</i> ₁ = <i>Printer</i> ₂

Figure 5: Direct attack correlation in the illegal file access scenario

4.1 Correlation definitions

Let E and F be two logical expressions having the form: $E = \text{expr}_{E_1}, \text{expr}_{E_2}, \dots, \text{expr}_{E_m}$, and $F = \text{expr}_{F_1}, \text{expr}_{F_2}, \dots, \text{expr}_{F_n}$. Each expr_i in E and F is either a predicate or a negation of predicate (not equivalent to *true*), namely expr_i must have one of the following forms: $\text{expr}_i = \text{pred}$, or $\text{expr}_i = \text{not}(\text{pred})$.

Definition 1: Correlation We say that logical expressions E and F are correlated if there exist i in $[1, m]$ and j in $[1, n]$ such that expr_{E_i} and expr_{F_j} are unifiable through a most general unifier (mgu) Θ .

For instance, the post condition of action *touch*(*Agent*, *File*) is correlated with the pre condition of action *remove*(*Agent*, *File*). This is because these two logical expressions have in common predicate *owner*(*Agent*, *File*). After renaming the variables of *owner*(*Agent*, *File*) that respectively appear in the post condition of action *touch* and the pre condition of action *remove* into *owner*(*Agent*₁, *File*₁) and *owner*(*Agent*₂, *File*₂), we can conclude that these expressions are unifiable through mgu Θ such that $\text{Agent}_1 = \text{Agent}_2$ and $\text{File}_1 = \text{File}_2$.

Definition 2: Direct attack correlation We say that attacks A and B are directly correlated if expressions $\text{Post}(A)$ and $\text{Pre}(B)$ are correlated.

Intuitively, correlation between attacks A and B means that A enables the intruder to then perform attack B . Figure 5 shows attacks that are directly correlated in the *illegal file access* scenario.

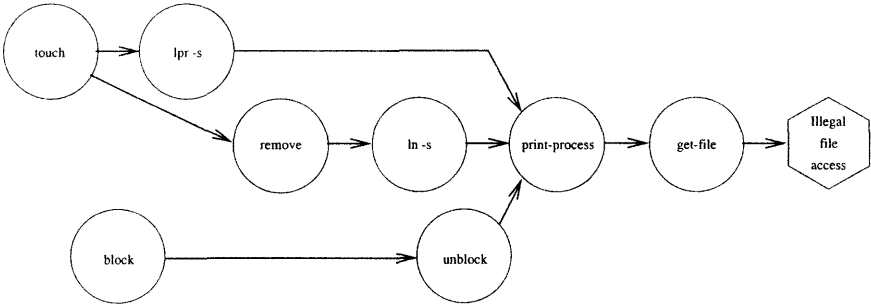


Figure 6: Illegal file access scenario

Definition 3: Indirect attack correlation We say that attacks A and B are indirectly correlated through domain rules R_1, \dots, R_n if: (1) $Post(A)$ is correlated with $Pre(R_1)$ through a mgu Θ_0 , and (2) For each j in $[1, n - 1]$, $Post(R_j)$ is correlated with $Pre(R_{j+1})$ through a mgu Θ_j , and (3) $Post(R_n)$ is correlated with attack $Pre(B)$ through a mgu Θ_n .

For instance, it is easy to verify that attack $touch(Agent, File)$ is indirectly correlated with attack $lpr-s(Agent, Printer, File)$ through the domain rule $owner_right(File)$.

Definition 4: Direct objective correlation We say that attack A is directly correlated to intrusion objective O if expressions $Post(A)$ and $State_condition(O)$ are correlated.

Definition 5: Indirect objective correlation Same definition as definition 5 by replacing $State_condition(O)$ for $Pre(B)$.

Intuitively, correlation between attack A and intrusion objective O means that attack A enables the intruder to achieve objective O . For instance, attack $get-file(Agent, File)$ is directly correlated with intrusion objective $illegal_file_access(File)$. Figure 6 shows the result of applying the correlation definitions to the *illegal file access scenario*.

4.2 Generating correlation rules

In [CM02], we show how to automatically generate *correlation rules*. Due to space limitation, we shall simply give the intuition here.

An attack correlation rule enables the correlation process to correlate two alerts corresponding to correlated attacks. For instance, attack $remove(Agent_1, File_1)$ is correlated to attack $ln-s(Agent_2, Link_2, File_2)$ when $File_1 = Link_2$. In this case, the associated correlation rule will say that an alert $Alert_1$ corresponding to detection of attack $remove$ can be correlated with an alert $Alert_2$ corresponding to detection of attack $ln-s$ if the target file associated with $Alert_1$ is equal to the target link associated with $Alert_2$. Of course, an implicit condition to correlate $Alert_1$ with $Alert_2$ is that the attack associated with $Alert_1$ occurred *before* the attack associated with $Alert_2$.

We similarly generate objective correlation rules to correlate an alert with an intrusion objective. When the correlation process receives an alert and there is a correlation rule that applies to correlate this alert to an intrusion objective, the correlation process will check if this objective is achieved or not. For instance, attack $winnuke$ is correlated with objective DOS_on_DNS . If an alert corresponding to attack $winnuke$ on a given *Host* is received,

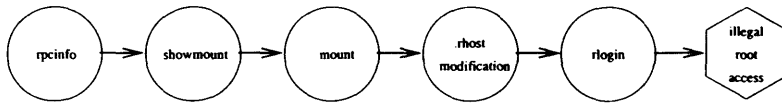


Figure 7: Illegal root access scenario

then the correlation process will check if *Host* corresponds to a DNS server. Notice that this data is generally not provided in the alert. Therefore, we need to combine “classical” IDS with other tools that collect additional information about the monitored system such as its topology and configuration. This problem is briefly discussed in the conclusion.

5 Hypothesis generation in the correlation process

In the correlation process, hypothesis generation (HG) is used in two different situations:

1. When all the steps of a given intrusion scenario are detected, the correlation process will succeed in tracking the intruder by reconstituting his intrusion scenario. However, it may happen that some steps in an intrusion scenario are not detected by any of the IDS. In this case, HG will try to generate minimal hypotheses corresponding to undetected attacks in order to complete the intrusion scenario detection. We suggest raising a *virtual alert* for each hypothesis successfully generated.
2. When the correlation process succeeds in correlating several attacks but no intrusion objective is achieved yet, HG is used to generate an intrusion objective consistent with these attacks. This is used by the correlation process to anticipate over the intruder’s intention in order to prepare the most adequate reaction.

5.1 Virtual alerts

The correlation process will attempt to create *virtual alerts* when it is plausible that some attacks are not detected. That is, when two alerts cannot be correlated, we try to insert one or several virtual alerts between them. Let us describe the two main steps of the virtual alert creation function. Let us assume that *Alert*₁ and *Alert*₂ are not correlated.

- Let *Attack*₁ and *Attack*₂ be the attacks respectively associated with *Alert*₁ and *Alert*₂. The correlation process will then attempt to find a path of attacks to connect *Attack*₁ with *Attack*₂. Of course, this path must be formed by attacks that might be not detected by any IDS that provides alerts to the correlation process.

For example, let us assume that, in the illegal root access scenario¹ (see figure 7), the modification of the *.rhost* file is not detected. In this case, the correlation process receives the *rlogin* alert without being able to correlate it with the *mount* alert. However, the correlation process knows that attack *mount* can be correlated with attack *rlogin* through attack *.rhost*.

¹Due to space limitation, we do not give a complete specification of various actions and intrusion objective included in this scenario (see [CM02] for a more complete discussion of this scenario).

- The second step of the function replaces the path of attacks by a path of virtual alerts by instantiating each attack. From the first attack we create a first virtual alert. This virtual alert is correlated with *Alert₁*. We do the same for the next attacks of the path until *Alert₂* is achieved. At this point, the correlation process verifies whether it is possible to correlate the last virtual alert with *Alert₂*.

In the last example, we had a single attack in our path corresponding to the *.rhost* modification. We create a virtual alert associated with this attack. According to the correlation rules between *mount* alert and *.rhost* modification alert, the target IP addresses must be the same. Consequently, the virtual alert is initialised with the target IP address of the *mount* alert. We then check correlation between the virtual alert *.rhost* and the *rlogin* alert. This test could fail if the target IP addresses of these two alerts are not equal.

5.2 Derivation of intrusion objective

When the correlation process has detected the beginning of a scenario, it tries to find out what will be the next steps that might enable the intruder to achieve an intrusion objective.

For this purpose, the correlation process applies an approach similar to the first step used to raise virtual alerts. It analyses the possible correlations between attacks and between an attack and an intrusion objective to find a path of attacks between the last detected alert of the scenario and an intrusion objective. When this alert can be connected to different intrusion objectives through different paths, our strategy is simply to select an intrusion objective that corresponds to the shortest path.

6 Discussions and examples of scenario detection

In the intrusion detection context, the intruder whose plans are being inferred is not cooperative and observations are gleaned through IDS alerts. This point and the computer security context bring to light several issues to take in consideration. The objective of this section is to show, through the intrusion scenarios introduced in section 2, how our approach addresses these issues:

- Unobserved actions: There are multiple reasons that can make an attack unobservable. Signature based IDS are not able to recognize unknown exploits and even variations of known exploits can make them undetectable. Furthermore there can be holes in the IDS network coverage that make impossible detection of some malicious attacks. Our approach to solve the problem that some steps in an intrusion scenario are not detected is based on hypothesis generation as shown in section 5.
- Optional actions: Figure 8 shows detection of intrusion scenario *DOS_on_DNS* when the intruder performs the sequence *nslookup*, *ping*, *scan*, *winnuke*. Actually, actions *ping* and *scan* are optional since the intruder may directly attempt the *winnuke* attack without checking that the server is active (with the *ping* command) and Windows is installed (with a *scan* of port 139). Representing intrusion scenario that includes optional actions is immediate in our approach. If the intruder does not execute *ping* or *scan*, we shall simply detect a simpler scenario that does not include these actions.

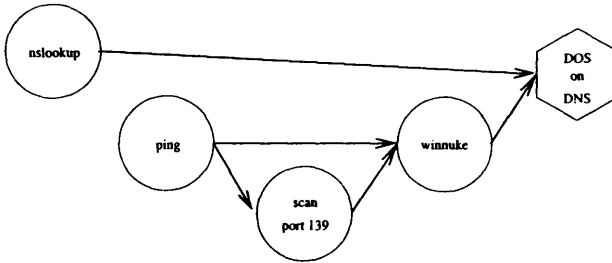


Figure 8: DOS on DNS scenario

- Representation of intrusion scenario variations: As an intruder executes his plan, his actions may lead him to draw some conclusions i.e. gain some knowledge about the network or some host for example. This may lead in small variations in the execution of the intruder's plan. We have to be able to represent these variations to take them in consideration. For instance, in the *DOS_on_DNS* scenario, the intruder may prefer using *traceroute* instead of a *ping* to know which machines are active in the network he wants to attack. Representing intrusion scenario variations is straightforward in our approach. We have simply to specify the pre and post conditions of attack *traceroute* and the correlation process will automatically derive that replacing *traceroute* by *ping* also enables the intruder to achieve the *DOS_on_DNS* objective.
- Management of repeated observations: an intruder that will repeat several times the steps of an intrusion scenario will potentially activate the plan recognition several times. This may lead to an explosion of the number of alerts as in [GG01a]. For instance, if one specifies an intrusion scenario *Winnuke_Attack* by the sequence *ping*, *scan port 139*, *winnuke* and if the intruder executes 100 *pings* and 100 *scans* and then 1 *winnuke*, this may lead to $100 \times 100 = 10000$ detections of the *Winnuke_attack*. Our approach will generate only one alert for such a case, even if this alert corresponds to a complex scenario.

7 Conclusion

Based on the observation that an intrusion scenario might be represented as a planning activity, we suggest a model to recognize intrusion scenarios and malicious intentions. This model does not follow previous proposals that require to explicitly specify a library of intrusion scenarios. Instead, our approach is based on specification of elementary attacks and intrusion objectives. We then show how to derive correlation relations between two attacks or between an attack and an intrusion objective. Detection of complex intrusion scenario is obtained by combining these binary correlation relations.

Our approach is efficient to cluster repeated actions in a single scenario. We also suggest using hypothesis generation to recognize intrusion scenarios when some steps in these scenarios are not detected. We have implemented in Prolog the functions that perform attack and objective correlations in the CRIM prototype [Cup01, CM02]. Attacks are actually specified in Lambda [CO00], which is fully compatible with the attack model suggested in this paper and alerts are represented in the IDMEF format [CD01].

There are several issues to this work. When the intruder did not achieved his intrusion ob-

jective yet but there are several possible intrusion objectives consistent with a given sequence of correlated attacks, our current strategy is simply to select the objective that requires the shortest path of attacks to be achieved. Our course, it would be useful to significantly enhance this strategy. We are studying approaches based on Bayesian Networks to infer the best intrusion objectives that explain all the observations. As suggested in [GG01b], our solution should also be able to consider situations where the intruder has multiple goals.

Another point is that to decide if a given intrusion scenario is achieved or not, it is often necessary to combine information provided by “classical” IDS with other information about the system monitored by the IDS: its topology, configuration and other data about the type and version of the operating systems and installed applications. For instance, to decide if the objective *DOS_on_DNS* is achieved it is necessary to know on which system is installed the DNS server. This is not provided by classical IDS but other tools exist that may be used to collect it. Since current IDS also provide alerts that do not allow us to distinguish between successful or failing attacks, these additional data would be also useful for that purpose. This problem is currently investigated in the ongoing project DICO (see also [MMDD02]).

8 Acknowledgements

This work was partially funded by the DGA/CELAR/CASSI as a part of the Mirador project and then by the French Ministry of Research as part of the DICO project. The authors would like to thank all the members of these projects, especially the members of the sub-project “Correlation”: Hervé Debar, Ludovic Mé and Benjamin Morin.

References

- [CD01] D. Curry and H. Debar. Intrusion detection message exchange format data model and extensible markup language (xml) document type definition. draft-itetf-idwg-idmef-xml-06.txt, December 2001.
- [CM02] F. Cuppens and A. Miège. Alert Correlation in a Cooperative Intrusion Detection Framework. In *IEEE Symposium on Security and Privacy*, Oakland, USA, 2002.
- [CO00] F. Cuppens and R. Ortalo. Lambda: A language to model a database for detection of attacks. In *Third International Workshop on the Recent Advances in Intrusion Detection (RAID'2000)*, Toulouse, France, October 2000.
- [Cup01] F. Cuppens. Managing Alerts in a Multi-Intrusion Detection Environment. In *17th Annual Computer Security Applications Conference New-Orleans*, New-Orleans, USA, December 2001.
- [GG01a] C. Geib and R. Goldman. Plan Recognition in Intrusion Detection Systems. In *DARPA Information Survivability Conference and Exposition (DISCEX)*, June 2001.
- [GG01b] C. Geib and R. Goldman. Probabilistic Plan Recognition for Hostile Agents. In *Florida AI Research Symposium (FLAIR)*, Key-West, USA, 2001.
- [MMDD02] Benjamin Morin, Ludovic Mé, Hervé Debar, and Mireille Ducassé. M2D2: A Formal Data Model for IDS Alert Correlation. In *Fifth International Workshop on the Recent Advances in Intrusion Detection (RAID'2002)*, Zurich, October 2002.
- [VS01] A. Valdes and K. Skinner. Probabilistic Alert Correlation. In *Fourth International Workshop on the Recent Advances in Intrusion Detection (RAID'2001)*, Davis, USA, October 2001.
- [ZMB02] Jacob Zimmermann, Ludovic Mé, and Christophe Bidan. Introducing reference flow control for intrusion detection at the OS level. In *Fifth International Workshop on the Recent Advances in Intrusion Detection (RAID'2002)*, Zurich, October 2002.

A Self-growing Probabilistic Decision-based Neural Network with Applications to Anchor/Speaker Identification

S.S. Cheng, Y.H. Chen, C.L. Tseng, Hsin-Chia Fu[†], and H.T. Pao[‡]

Department of Computer Science and Information Engineering

[‡]*Department of Management Science*

National Chiao Tung University, Hsin-Chu, Taiwan, ROC

Abstract. In this paper, we propose a new learning algorithm for a mixture Gaussian based neural network, called Self-growing Probabilistic decision-based neural networks (SPDNN) for better density function estimation, and pattern classification. We also developed a new Self-growing Mixture Gaussian learning (SMGL) algorithm, that is able to find the natural number of components based on a self-growing validity measure, Bayesian Information Criterion (BIC). It starts with a single component randomly initialized in the feature space and grows adaptively during the learning process until most appropriate number of components are found. In our experiments on anchor/speaker identification, we have observed noticeable improvement among various model-based or vector quantization-based classification schemes. *Key Words:* Self-growing Probabilistic Decision-based Neural Networks (SPDNN), Supervised learning, Competitive learning, Unsupervised learning, Validity measure.

1 Introduction

The last two decades have seen a growing number of researcher and practitioner apply neural networks (NNs) to a variety of problems such as pattern classification and function approximation. In many application, data clustering techniques have been applied to discover and to extract hidden structure in a data set. Thus the structural relationships between individual data points can be detected. In general, clustering is an unsupervised learning process [1, 5].

A variety of competitive learning schemes have been developed, distinguishing in their approaches to competition and learning rules. The simplest and most prototypical CL algorithms are mainly based on the winner-take-all (WTA) [4] (or hard competitive learning) paradigm, where adaption is restricted to the winner that is the single neuron prototype best matching the input pattern. Different algorithms in this paradigm such as LBG (or generalized Lloyd) [9, 2, 10] and k-Means [11] have been well recognized.

There are two major issues associated with conventional competitive learning, namely, sensitivity to initialization and difficulty in determining the number of prototypes. In general, selecting the appropriate number of prototypes is a difficult task, as we do not usually know the number of clusters in the input data a priori. It is therefore desirable to develop an algorithm that has no dependency on the initial prototype locations and is able to adaptively generate prototypes to fit the input data patterns.

In this paper, we propose a new learning algorithm for a mixture Gaussian based neural network, called Self-growing Probabilistic decision-based neural networks (SPDNN) for better estimate density functions corresponding to different pattern classes. We also developed a new Self-growing Mixture Gaussian learning (SMGL) method, that is able to find the natural number of components based on a self-growing validity measure, Bayesian Information Criterion (BIC). It starts with a single component randomly initialized in the feature space and grows adaptively during the learning process until most appropriate number of components are found. In our experiments on anchor/speaker identification, we have observed noticeable improvement among various model-based or vector quantization-based classification schemes.

The remainder of this paper is organized as follows. In Section 2, we describe in detail the architecture of SPDNN and the SMGL Algorithm. Section 3 presents the experimental results on clustering analysis and anchor/speaker identification. Finally, Section 4 gives the summary and concluding remarks.

2 Self-growing Probabilistic Decision-based Neural Network

As shown in Figure 1, Self-growing Probabilistic Decision-based Neural Network (SPDNN)[3] is a multi-variate Gaussian neural network [6, 8]. The training scheme of SPDNN is based on the so-called *LUGS* (Locally Unsupervised Globally Supervised) learning. There are two phases in this scheme: during the locally-unsupervised (LU) phase, each subnet is trained individually by the proposed unsupervised competitive learning algorithm (SGML) (see Section 2.2.5), and no mutual information across the classes may be utilized. After the LU phase is completed, the training enters the Globally-Supervised (GS) phase. In GS phase teacher information is introduced to reinforce or anti-reinforce the decision boundaries between classes. The discriminant functions in all clusters will be trained by the two-phase learning. The detailed description of PDBNN detector is given in the following. Detail description of the SPDNN model will be given in the following sections.

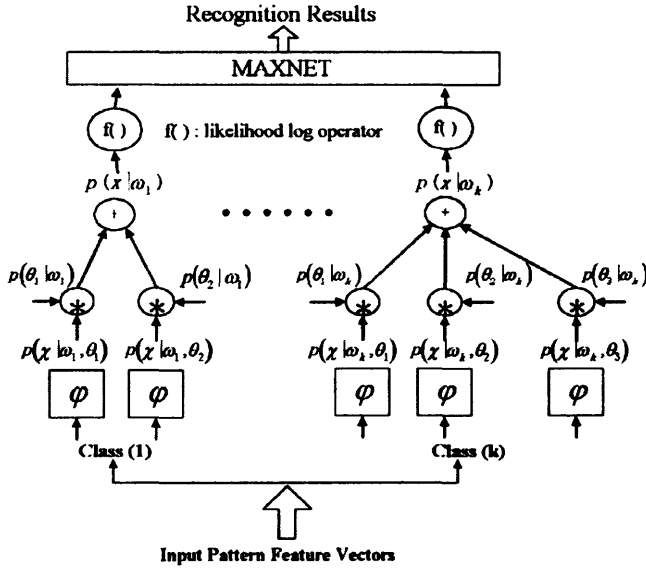
2.1 Discriminant Functions of SPDNN

One of the major differences between multi-variate Gaussian neural networks (MGNN) [6, 8] and SPDNN is that SPDNN extends the fixed number of clusters in a subnet of MGNN to a flexible number of clusters in a subnet of the SPDNN. That is, the subnet discriminant functions of SPDNN are designed to model the log-likelihood functions of different complexed pixel distributions of a data pattern. Given a set of iid patterns $\mathbf{X}^+ = \{x(t); t = 1, 2, \dots, N\}$, we assume that the likelihood function $p(\mathbf{x}(t) | \omega_i)$ for class ω_i is a mixture of Gaussian distributions.

Define $p(x(t) | \omega_i, \Theta_{r_i})$ as one of the Gaussian distributions which comprise $p(x(t) | \omega_i)$, where Θ_{r_i} represents the parameter set $\{\mu_{r_i}, \Sigma_{r_i}\}$ for a cluster r_i in a subnet i :

$$p(\mathbf{x}(t) | \omega_i) = \sum_{r_i=1}^{R_i} P(\Theta_{r_i} | \omega_i) p(\mathbf{x}(t) | \omega_i, \Theta_{r_i}),$$

where $P(\Theta_{r_i} | \omega_i)$ denotes the prior probability of the cluster r_i . By definition, $\sum_{r_i=1}^{R_i} P(\Theta_{r_i} | \omega_i) = 1$, where R_i is the number of clusters in ω_i .

Figure 1: The schematic diagram of a k -class SPDNN.

The discriminate function of the multi-class SPDNN models the log-likelihood function

$$\begin{aligned} \varphi(\mathbf{x}(t), \mathbf{w}_i) &= \log p(\mathbf{x}(t) | \omega_i) \\ &= \log \left[\sum_{r_i=1}^{R_i} P(\Theta_{r_i} | \omega_i) p(\mathbf{x}(t) | \omega_i, \Theta_{r_i}) \right], \end{aligned} \quad (1)$$

where $\mathbf{w}_i = \{\mu_{r_i}, \Sigma_{r_i}, P(\Theta_{r_i} | \omega_i), T_i\}$. T_i is the output threshold of the subnet i .

In most general formulations, the basis function of a cluster should be able to approximate the Gaussian distribution with a full rank covariance matrix, i.e., $\varphi(\mathbf{x}, \omega_i) = -\frac{1}{2} \mathbf{x}^T \Sigma_{r_i}^{-1} \mathbf{x}$, where Σ_{r_i} is the covariance matrix. However, for applications which deal with high-dimension data but a finite number of training patterns, the training performance and storage space requirements discourage such matrix modeling. A natural simplifying assumption is to assume uncorrelated features of unequal importance. That is, suppose that $p(\mathbf{x}(t) | \omega_i, \Theta_{r_i})$ is a D -dimensional Gaussian distribution with uncorrelated features:

$$\begin{aligned} p(\mathbf{x}(t) | \omega_i, \Theta_{r_i}) &= \frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma_{r_i}|^{\frac{1}{2}}} \\ &\cdot \exp \left[-\frac{1}{2} \sum_{d=1}^D \frac{(x_d(t) - \mu_{r_i,d})^2}{\sigma_{r_i,d}^2} \right], \end{aligned} \quad (2)$$

where $\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_D(t)]^T$ is the input, $\mu_{r_i} = [\mu_{r_i,1}, \mu_{r_i,2}, \dots, \mu_{r_i,D}]^T$ is the mean vector, and diagonal matrix $\Sigma_{r_i} = \text{diag}[\sigma_{r_i,1}^2, \sigma_{r_i,2}^2, \dots, \sigma_{r_i,D}^2]$ is the covariance matrix. As shown in Figure 1, an SPDNN contains K subnets which are used to represent a K -category classification problem. Inside each subnet, an elliptic basis function (EBF) serves as

the basis function for each cluster r_i :

$$\varphi(\mathbf{x}(t), \omega_i, \Theta_{r_i}) = -\frac{1}{2} \sum_{d=1}^D \alpha_{r_i,d} (x_d(t) - \mu_{r_i,d})^2 + \theta_{r_i} \quad (3)$$

where $\theta_{r_i} = -\frac{D}{2} \ln 2\pi + \frac{1}{2} \sum_{d=1}^D \ln \alpha_{r_i,d}$. After passing an exponential activation function, $\exp\{\varphi(\mathbf{x}(t), \omega_i, \Theta_{r_i})\}$ can be viewed as a Gaussian distribution, as described in (2), except for a minor notational change: $\frac{1}{\alpha_{r_i,d}} = \sigma_{r_i,d}^2$.

2.2 Learning Rules for SPDNN

2.2.1 Global supervised learning

Since the number of clusters in a subnet of an SPDNN can be adjusted in the self-growing phase, each subnet is initialized with one cluster. Suppose that $\mathbf{X}_i^+ = \{\mathbf{x}_i(1), \dots, \mathbf{x}_i(M_i)\}$ is a set of given training patterns, which correspond to one of the L classes $\{\omega_i, i = 1, \dots, L\}$; the mean μ_i and covariance Σ_i of the first cluster in subnet i can be initialized as

$$\mu_i = \frac{1}{M_i} \sum_{m=1}^{M_i} \mathbf{x}_i(m), \quad (4)$$

$$\Sigma_i = \frac{1}{M_i - 1} \sum_{m=1}^{M_i} (\mathbf{x}_i(m) - \mu_i)(\mathbf{x}_i(m) - \mu_i)^T. \quad (5)$$

During the **supervised learning** phase, training data are used to fine tune the decision boundaries of each class. Each class is modeled by a subnet with discriminant functions, $\varphi(\mathbf{x}(t), \mathbf{w}_i)$, $i = 1, 2, \dots, L$. At the beginning of each supervised learning phase, use the still-being-trained SPDNN to classify all the training patterns $\mathbf{X}_i^+ = \{\mathbf{x}_i(1), \mathbf{x}_i(2), \dots, \mathbf{x}_i(M_i)\}$ for $i = 1, \dots, L$. $\mathbf{x}_i(m)$ is put into class ω_i if $\varphi(\mathbf{x}_i(m), \mathbf{w}_i) > \varphi(\mathbf{x}_i(m), \mathbf{w}_k)$, $\forall k \neq i$, and $\varphi(\mathbf{x}_i(m), \mathbf{w}_i) \geq T_i$, where T_i is the output threshold for subnet i . According to the classification results, the training patterns for each class i can be divided into three subsets:

- $D_1^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \in \omega_i, \mathbf{x}_i(m) \text{ is classified into } \omega_i \text{ (the correctly classified set)}\}$;
- $D_2^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \in \omega_i, \mathbf{x}_i(m) \text{ is misclassified into another class } \omega_j \text{ (the false rejection set)}\}$;
- $D_3^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \notin \omega_i, \mathbf{x}_i(m) \text{ is misclassified into class } \omega_i \text{ (the false acceptance set)}\}$.

The following reinforced and antireinforced learning rules [8] are applied to the corresponding misclassified subnets.

Reinforced Learning :

$$\mathbf{w}_i^{(m+1)} = \mathbf{w}_i^{(m)} + \eta \nabla \varphi(\mathbf{x}_i(m), \mathbf{w}_i). \quad (6)$$

Antireinforced Learning :

$$\mathbf{w}_j^{(m+1)} = \mathbf{w}_j^{(m)} - \eta \nabla \varphi(\mathbf{x}_i(m), \mathbf{w}_j). \quad (7)$$

In (6) and (7), η is a user defined learning rate, where $0 < \eta \leq 1$. For the data set D_2^i , reinforced and antireinforced learning are applied to classes ω_i and ω_j , respectively. As for

the false acceptance set D_j^i , antireinforced learning is applied to class ω_i , and reinforced learning is applied to class ω_j , to which $x_i(m)$ belongs. The gradient vectors $\nabla\varphi$ in (6) and (7) can be computed in a similar manner, as proposed in [8].

Threshold Updating The threshold value T_i of a subnet i in the SPDNN recognizer can also be learned by means of reinforced or antireinforced learning rules.

2.2.2 Self-growing a new cluster

The network enters the self-growing phase when the supervised learning reaches a saturated learning state but with unsatisfactory classification accuracy. In other words, the whole training set has been presented a few times, the train status (especially the recognition accuracy) remains unchanged or unimproved, i.e., the training process falls into a local minimum. To escape from the local minimum, an extra cluster is needed in order to reshape the energy distribution of the current SPDNN.

We have developed a new Self-growing Mixture Gaussian Learning (SMGL) algorithm that is able to find appropriate number of clusters based on a self-growing validity measure, Bayesian Information Criterion (BIC).

2.2.3 Bayesian Information Criterion (BIC)

One advantage of the mixture-model approach to clustering is that it allows the use of approximate Bayes factors to compare models. This gives a means of selecting not only the parameterization of the model, the clustering method, but also the number of clusters. The Bayes factor is the posterior odds for one model against the other assuming neither is favored a priori. This paper proposes an iteration method based on the Bayesian factor to determine the number of clusters in hierarchical clustering based on the classification likelihood. In the followings, we will describe an approximation method to derive the Bayes factor, called Bayesian Information Criterion (BIC).

Given a set of patterns $\mathbf{X}^+ = \{\mathbf{x}(t); t = 1, 2, \dots, N\}$ and a set of candidate models $\mathcal{M} = \{M_i \mid i = 1, \dots, L\}$, each model associated with a parameter set \mathbf{w}_i . If we want to select a proper model M_i from \mathcal{M} to represent the distribution of \mathbf{X}^+ , the posterior probability can be used to make the decision. Assuming a prior distribution $P(\mathbf{w}_{r_i} \mid M_i)$ for the parameters of each model M_i , the posterior probability of a model M_i is

$$\begin{aligned} P(M_i \mid \mathbf{X}^+) &= \frac{P(M_i)p(\mathbf{X}^+ \mid M_i)}{p(\mathbf{X}^+)} \\ &\propto P(M_i) \cdot p(\mathbf{X}^+ \mid M_i) \\ &\propto P(M_i) \cdot \sum_{r_i} p(\mathbf{X}^+ \mid \mathbf{w}_{r_i}, M_i) P(\mathbf{w}_{r_i} \mid M_i), \end{aligned}$$

where \mathbf{w}_{r_i} is the parameter set of cluster r_i in class i . If each model is equally likely a prior, then $P(M_i \mid \mathbf{X}^+)$ is proportional to the posterior probability that the data conform to the model M_i . Ripley proposes [13, 12] that the calculation of posterior probability $P(\mathbf{X}^+ \mid M_i)$ can be achieved by the Laplace approximation, that gives

$$\log p(\mathbf{X}^+ \mid M_i) = \log p(\mathbf{X}^+ \mid \hat{\mathbf{w}}_{r_i}, M_i) - \frac{1}{2}d(M_i) \cdot \log N + O(1) \equiv \frac{1}{2}BIC(M_i, \mathbf{X}^+),$$

where $\hat{\mathbf{w}}_{r_i}$ is a maximum likelihood estimate, $d(M_i)$ is the number of free parameters in model M_i , and N is the number of train data. Accordingly, the larger the value of BIC, the stronger the evidence for the model. The BIC can be used to compare models with differing parameterizations, differing numbers of cluster components, or both.

2.2.4 Model selection by BIC

If there are two candidate models M_1 and M_2 for modeling a data set \mathbf{X}^+ , Raftery et al., suggest that the BIC difference ΔBIC_{21} [7] :

$$\Delta BIC_{21} = BIC(M_2, \mathbf{X}^+) - BIC(M_1, \mathbf{X}^+)$$

can be used to evaluate which model is a preferred one. Table 1 depicts the grade of evidence corresponding the values of the Bayesian difference for favoring M_2 against M_1 .

Table 1: The BIC difference, the posterior probability of M_2 , and the *grades of evidence* corresponding to favoring M_2 against M_1 .

BIC difference	$pr(M_2 \mathbf{X}^+)(\%)$	Evidence
0-2	50-70	Weak
2-6	75-95	Positive
6-10	95-99	Strong
> 10	> 99	Decisive

2.2.5 BIC-based Self-growing Mixture Gaussian Learning

There are three main aspects with respect to self-growing rules:

- I1: *When should a cluster be splitted?*
- I2: *Which cluster should be split into a pair of new gaussian components?*
- I3: *How many clusters are enough for representing a density distribution?*

On Issue I1, when the whole training dataset has been presented for a few times, the training status (especially the recognition accuracy) remains unchanged or unimproved. An extra cluster is suggested to improve the representation power of the SPDNN.

On Issue I2, suppose each cluster is splitted to calculate its value of ΔBIC_{21} . We grow one new Gaussian component from the cluster with highest value of ΔBIC_{21} which is large than the corresponding *growing – confidence*. From the viewpoint of BIC, the higher the ΔBIC_{21} has, the more confidence of the GMM_2 has.

On Issue I3, a splitting process terminates when the value of ΔBIC_{21} for each cluster is no longer larger than its previous value or the growing-confidence. In other words, the ΔBIC_{21} does not favor any of the splitted clusters.

Detail computing processes are depicted in the following SMGL algorithm. Before going into any further, we would like to define some notations:

- $BIC(\mathbf{X}^+, GMM_i)$: The BIC value of mixture Gaussian model i with data set \mathbf{X}^+ .
- $\Delta BIC_{21}(\mathbf{X}^+)$: $BIC(\mathbf{X}^+, GMM_2) - BIC(\mathbf{X}^+, GMM_1)$.

- *Growing confidence*: When a two-Gaussian model is to be created by splitting a uni-Gaussian model, the new model is preferred, if (1) the BIC value of new model, i.e., $BIC(\mathbf{X}^+, GMM_2)$ is larger than the BIC value of old model $BIC(\mathbf{X}^+, GMM_1)$, and (2) if $\Delta BIC_{21}(\mathbf{X}^+)$ is larger than a confidence threshold, which is called the *growing confidence*.

Self-growing Mixture Gaussian Learning algorithm

Input data set: $\mathbf{X}^+ = \{\mathbf{x}(t) : t = 1, \dots, N\}$

Parameter set for model M_i : $\mathbf{w}_i = \{P(\Theta_{r_i} | \omega_i), \Theta_{r_i} | r_i = 1, 2, \dots, compoNum\}$, where $P(\Theta_{r_i} | M_i)$ is the prior probability of the r_i th Gaussian component in class i , and $\Theta_{r_i} = \{\mu_{r_i}, \Sigma_{r_i}\}$ are the mean and covariance matrix of the r_i -th Gaussian component

BEGIN

$compoNum = 1$;

$\mathbf{w}_i = \{\mu_{compoNum}, \Sigma_{compoNum}\}$;

$BIC_set(compoNum) = 0$;

%% Initializing $BIC_set(compoNum)$; the BIC value of $GMM_{compoNum}$

$GMM_set(compoNum) = \mathbf{0}$;

%% Initializing $GMM_set(compoNum)$; the parameter set of a $GMM_{compoNum}$

1. If $\Delta BIC_{21}(\mathbf{X}^+) < \text{growing} - \text{confidence}$, then
initializing \mathbf{w}_i = the parameters of a uni-Gaussian model, with mean and variance computed from \mathbf{X}^+ ;
2. Data Clustering:
 $EM_cluster_j = \phi, j = 1, 2, \dots, compoNum$;
 for each pattern $\mathbf{x}(t)$:
 $c = \arg \max_r \{p(\Theta_{r_i} | \omega_i, \mathbf{x}(t))\}$; $EM_cluster_c = EM_cluster_c \cup \mathbf{x}(t)$;
3. Growing one component:
 Let $local\Delta BIC = \max_i \{\Delta BIC_{21}(EM_cluster_i)\}$;
 whichGrow = $\arg \max_i \{\Delta BIC_{21}(EM_cluster_i)\}$.
 If $local\Delta BIC < \text{growing} - \text{confidence}$, then
 Let $m = \arg \max_i \{BIC_set(i)\}$; $\mathbf{w}_i = GMM_set(m)$; goto END;
 else
 Let the new cluster be GMM_2 (which corresponds to the largest $BIC(GMM_2, EM_cluster_{whichGrow})$), and
 let the parameters of the two new clusters are initialized as:
 $\bar{\lambda}_1 = \{P(\bar{\Theta}_1 | \omega_i), \bar{\Theta}_1\}$, $\bar{\lambda}_2 = \{P(\bar{\Theta}_2 | \omega_i), \bar{\Theta}_2\}$, where
 $\bar{\Theta}_{r_i} = \{\bar{\mu}_{r_i}, \bar{\Sigma}_{r_i}\}$, for $r_i = 1, 2$;
 $P(\bar{\Theta}_1 | \omega_i) = P(\bar{\Theta}_2 | \omega_i) = \frac{1}{2}P(\Theta_{whichGrow} | \omega_i)$,
 $\mathbf{w}_i = \mathbf{w}_i \setminus \{P(\Theta_{whichGrow} | \omega_i), \Theta_{whichGrow}\}$;
 $\mathbf{w}_i = \mathbf{w}_i \cup \{\bar{\lambda}_1, \bar{\lambda}_2\}$; $compoNum = compoNum + 1$;
4. Global EM learning:
 Perform EM learning on all the clusters, and
 let $BIC_set(compoNum) = BIC(GMM_{compoNum}, \mathbf{X}^+)$;
 $GMM_set(compoNum) = \mathbf{w}_i$; Goto 2.

END

3 Experimental results

In this section, experimental results are presented in two parts. In the first part we use an synthetic data set and the second part explores the ability of SPDNN for anchor/speaker identification. In each part we perform EM based learning with six different initialization methods for the training of an SPDNN. These six different initialization methods are briefly explained in the followings. (1) RegularEM method: Using EM learning method with initial *Gaussian mean* values determined by randomly selected seeds from training data. (2) Model-Kmeans method: Using EM learning method with initial *Gaussian mean* values determined by K-means clustering. (3) Model-singleLink method: Using EM learning method with initial *Gaussian mean* values determined by single-link hierarchical clustering. (4) Model-averageLink method: Using EM learning method with initial *Gaussian mean* values determined by average-link hierarchical clustering. (5) Model-completeLink method: Using EM learning method with initial *Gaussian mean* values determined by complete-link hierarchical clustering. (6) Self-growing method: Using EM learning method with initial *Gaussian mean* values determined by the proposed BIC-based self-growing method.

Experiment 1: Synthetic data set drawn from a distribution of six mixture Gaussian clusters

The synthetic data set contains 600 data points, which are evenly divided into six Gaussian clusters. Figure 2 depicts the self-growing and *EM* learning processes from one up to six clusters. In this experiment, the number of mixture Gaussian clusters is limited to an upper bound of 10. As shown in Figure 3, the learning curve of the proposed method rises to its peak value when the number of cluster reaches 6, which is the predefined number of clusters in the data set.

Experiment 2: Anchor/speaker identification

The evaluation of the anchor/speaker identification experiments were conducted in the following manner. Speech data were collected from 19 female and 3 male anchor/speakers, of the evening TV news broadcasting programs in Taiwan. For each speaker, there are 180 TV news briefing of approximately 25 minutes sampling over 6 months. The speech data are partitioned into 5 second segments, which corresponds to 420 features vectors (mfcc). Each speaker was modeled by a SPDNN trained by three different amount of speech data (30, 60 and 90 seconds). Testing were done using the rest of speech data. Each segment of 5 second speech data was treated as a separate test utterance.

The experiments were primarily conducted to investigates

1. the capability of the proposed BIC based self-growing mixture Gaussian learning (SMGL) algorithm in determining the number of components in a mixture model;
2. the performance of the SPDNN for real problems e.g., speaker identification.

There is no theoretical way to estimate the number of mixture components *a priori*. For speaker modeling, the objective is to choose the minimum of components necessary to adequately model a speaker for good speaker identification. Figure 4 shows, the learning curve of (BIC values) versus the number of Gaussian components used in building a SPDNN speaker model. There are several observations to be made from these results. First, the sharp increase in the BIC values from 1 to 8 mixture components, and leveling off above 16 components.

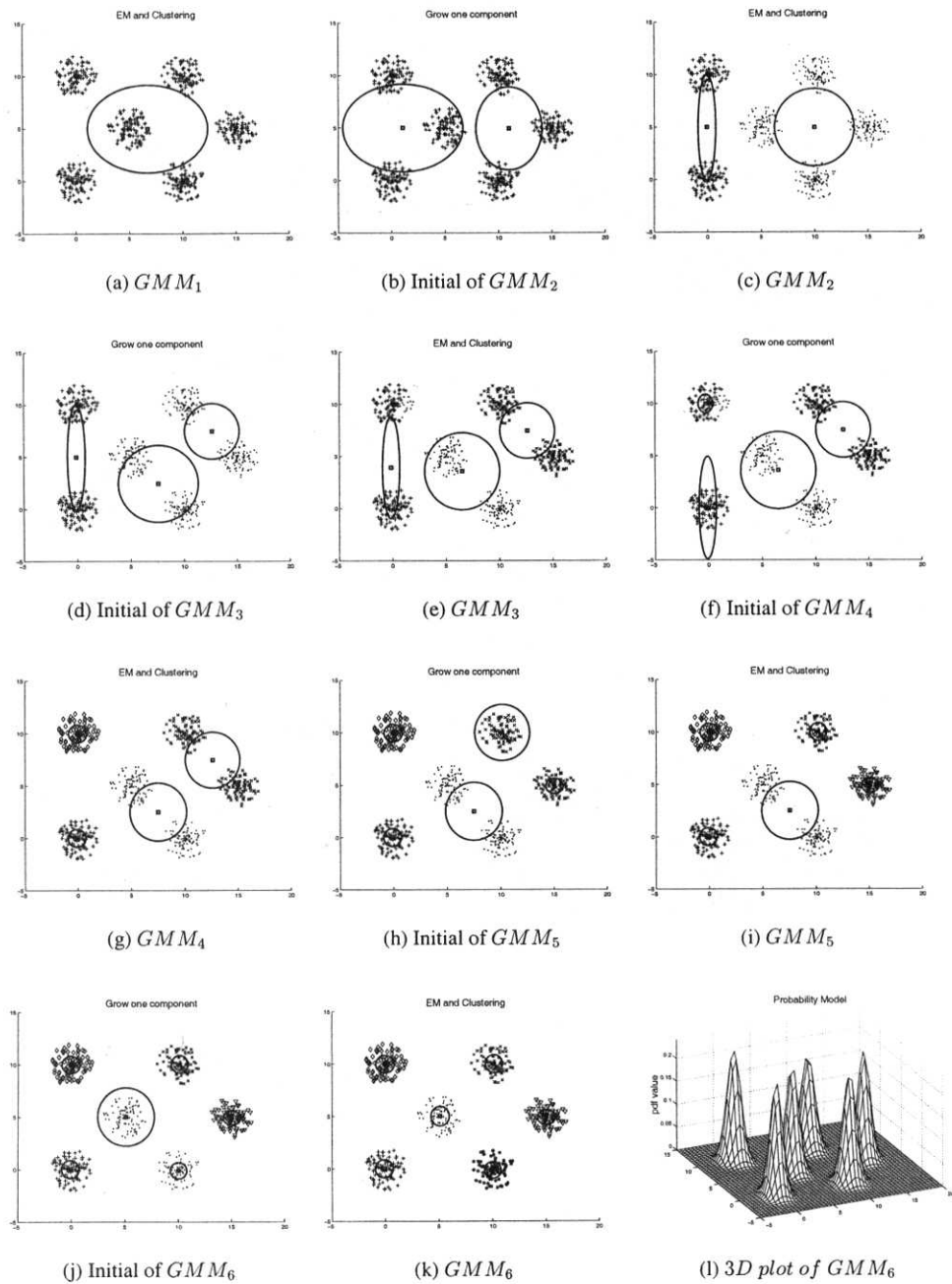


Figure 2: The learning process of self-growing with synthetic data.

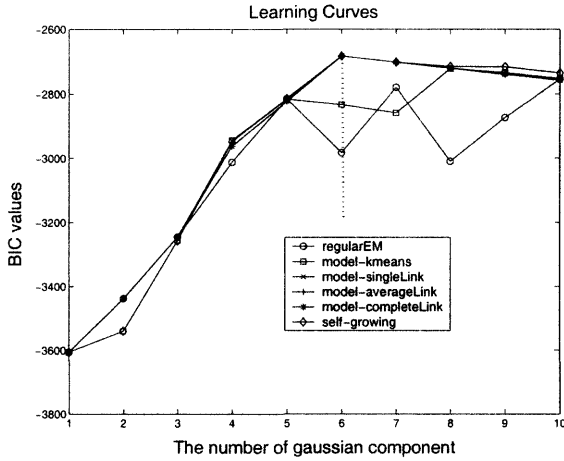


Figure 3: Learning curves of the six different methods applied on the synthetic data. The learning curve of the Self-growing method peaks at GMM_6 , it seems that the proposed SMGL method suggests a nature number of clusters for the synthetic data set.

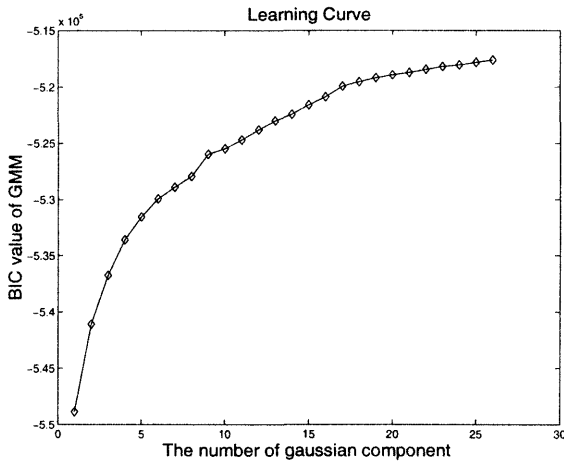


Figure 4: The learning curve of the SPDNN for an anchor/speaker identification system. The training data of each speaker's model is a 90 sec. of speech. The BIC values shows a sharp increase from 1 to 8 mixture components, and leveling off above 16 components.

Thus, the next set of experiments were performed by using

1. different amount (30, 60, and 90 seconds) of training speech data
2. different dimension of mfcc (mel-frequency features).

As shown in Table 2, by using different amount of training data and *mfcc* feature dimension, the number of mixture components corresponding to peak BIC values, ranges from 12 to 30. The performance evaluation was then computed as the percent of correctly identified segment

overall test utterance segments.

$$\% \text{ correct identification} = \frac{\# \text{ correctly identification segments}}{\text{Total \# of segments}} \times 100\%$$

The complete identification rates are also shown in Table 2. In the lower part of Table 2, the identification rates were tested from an SPDNN with fixed (32) components. It seems that the identification performance from self-growing SPDNN is slightly better than the fixed component model.

Table 2: Self-growing number of clusters and speaker identification performance for different amount of training speech data and dimension of *mfcc* feature vectors. In the body of the table, the first number is the mean value of the number of clusters, the second number after ‘/’ is the variance and the number in parentheses indicates the percentage (%) of identification performance.

Amount of Training speech	Dimension of <i>mfcc</i>			
	12	16	20	24
<i>Identification performance by Self-growing SPDNN</i>				
30s	12.32/1.45(89.51)	13.32/2.03(92.00)	14.84/2.54(94.24)	15.8/2.12(94.81)
60s	17.31/2.16(90.28)	19/1.19(95.08)	20.84/2.41(96.81)	23.05/2.80(96.95)
90s	20.32/2.67(93.44)	23.79/2.55(96.46)	25.94/2.97(97.78)	30.21/3.6(97.90)
<i>Identification performance by fixed component SPDNN</i>				
90s	32(92.57)	32(95.60)	32(97.39)	32(97.71)

4 Concluding Remarks

In this paper, we have presented the SMGL algorithm for the learning of SPDNN. The SMGL algorithm tries to tackle two long standing critical problems in clustering, namely, 1) the difficulty in determining the number of clusters, and 2) the sensitivity to prototype initialization. Our SMGL algorithm is based on a split validity criterion, Bayesian Information Criterion (BIC). Using SMGL for clustering data, we need to randomly initialize only one prototype in the feature space. During the learning process, according to the split validity criterion (BIC), one prototype is chosen to split into two prototypes. This splitting process terminates when the BIC values of each cluster reaches their highest points. We have conducted experiments on a variety of data types and demonstrated that the SMGL algorithm is indeed a powerful, effective, and flexible technique in classifying clusters. Recently we have successfully applied SMGL to TV news anchor/speaker identification. Since TV news speech data are highly dynamic, SMGL is able to adaptively split according the actual datasets present. In addition, features in TV news speech are usually high dimensional, SMGL has demonstrated its ability in dealing with such data.

Acknowledgment

The authors acknowledge Prof. S.Y. Kung, Dr. S.H. Lin for their helpful suggestions regarding the probabilistic DBNN, and statistical pattern recognition methods. This research was supported in part by the National Science Council under Grant NSC 90-2213-E009-047.

References

- [1] C. Dacastecker, "Competitive clustering," in *Proc. IEEE Int. Neural Networks Conf.*, vol. 2, 1988, p. 833.
- [2] E. Forgy, "Cluster Analysis of multivariate data: Efficiency vs. interpretability of classifications," in *Biometrics*, vol. 21, p. 768, 1965.
- [3] H. C. Fu, H. Y. Chang, Y. Y. Xu, and H. T. Pao, "User Adaptive Handwriting Recognition by Self-growing Probabilistic Decision-based Neural Networks," in *IEEE Transaction on Neural Networks*, Vol. 11, No.6, Nov. 2000.
- [4] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*. New York: Addison-Wesley, 1991.
- [5] J. M. Jolion, P. Meer, and S. Bataouche, "Robust clustering with applications in computer vision," in *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 791-802, Aug. 1991.
- [6] M. I. Jordan, and R. A. Jacobs, "Hierarchical mixture of experts and the EM Algorithm," in *Neural Computation*, 6, pp. 181-214, 1994.
- [7] R. E. Kass, "Bayes Factors, in *Journal of the American Statistical Association*," 90, pp.773-795, 1995.
- [8] Shang-Hung Lin, S. Y. Kung, and L. J. Lin, "Face Recognition/Detection by Probabilistic Decision-based Neural Networks," in *IEEE Transactions on Neural Networks, special issue on Artificial Neural Network and Pattern Recognition*, Vol. 8, No. 1, pp. 114-132, 1997.
- [9] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," in *IEEE Trans. Commun.*, vol. 28, pp. 84-85, 1980.
- [10] S. P. Lloyd, "Least squares quantization in PCM," in *IEEE Trans. Inform. Theory*, vol. 28, pp. 129-137, 1982.
- [11] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probability*. Berkeley, CA: Univ. California Press, 1967, pp. 281-297.
- [12] A. E. Raftery, *Bayesian Model Selection in Social Research*. University of Washington Demography Center Working paper no 94-12, September 1994.
- [13] B. D. Ripley, *Pattern Recognition and neural networks*. Cambridge University Press, 1996.

HyCAR – A Robust Hybrid Control Architecture for Autonomous Robots

Farlei J. Heinen, Fernando S. Osório

UNISINOS University – Universidade do Vale do Rio dos Sinos

Mestrado em Computação Aplicada - PIPCA

Av. Unisinos 950, São Leopoldo – RS - Brasil

{farlei, osorio} @exatas.unisinos.br

<http://ncg.unisinos.br/robotica/simulador>

Abstract. This work presents a new hybrid architecture applied to autonomous mobile robot control - HyCAR (Hybrid Control for Autonomous Robots). This architecture provides a robust control for robots as they become able to operate and adapt themselves to different environments and conditions. We designed this new hybrid control architecture, integrating the two main techniques used in robotic control (deliberative and reactive control) and the most important environment representation techniques (grids, geometric and topological maps), through a three-layer architecture approach (vital, functional and deliberative layers). To guarantee the robustness of our control system, we also integrated a localization module based on Monte Carlo localization method. This localization module possesses an important role in our control system, and supplies a solid base for the control and navigation of autonomous mobile robots. In order to validate our control architecture, a realistic simulator of mobile robots was implemented (SimRob3D) allowing the practical use of the proposed system. We implemented several three-dimensional environment models, as well as diverse sensorial and kinematics models found in actual robots. Our simulation results had demonstrated that the control system is perfectly able to determine the mobile robot position into a partially known environment, considering local or global localization, and also to determine if the robot needs to re-localize itself given an incorrect localization. In navigation tasks the robot was able to plan and follow self-generated trajectories in a dynamic environment, which can include several unexpected static and mobile obstacles. We also demonstrated that with the integration of topological and grid information we improved planning algorithm execution.

1. Introduction

The robotic mobile vehicles [1] are an interesting subject of research in the field of Artificial Intelligence, which tries to improve the autonomy and robustness of those vehicles. The main difference between mobile robots and other robotic research fields, such as the field related to robotic manipulators, is the fact of this kind of robotic application operates in complex environments that modify itself dynamically and usually are unpredictable. We are confronted to large scale environments due to robot locomotion capacity, and these environments can also enclose other static or mobile unknown obstacles. To operate in this kind of environments the robot must be able to acquire and use all available knowledge about the environment (map), to estimate the robot position in this environment, to possess the ability to recognize obstacles, and to answer in real time to different situations that can occur in this dynamic environment. Moreover, all these functionalities must operate in par-

allel, and then, modularity is indispensable. The tasks responsible for activities like to perceive the environment, to localize the robot in the environment, to plan trajectories and to move the robot preventing collisions, attempt to solve together the main problems treated in the study of the autonomous mobile robots.

The tasks that allow a mobile robot to move from a specified point to another point in the environment are called robotic navigation tasks. We are confronted to several problems that make the navigation in a real environment very complex: environments are dynamic and they can be modified as time passes (e.g. furniture can be moved from one position to another one); changes in the environment can obligate to change the initial task planning; mobile obstacles with unpredictable exact trajectories can be found moving continuously around the environment (e.g. walking humans); data from sensors can be inexact and subject to errors, as well as, commands sent to actuators can be incorrectly executed; among others problems. All these problems demand robots that are able to deal with dynamic and uncertain data.

The first main problem we need to solve in autonomous mobile robot navigation is the problem of robot localization. Without knowing the actual position of the robot related to the known map of the environment (some initial environment representation), a control system has a lot of difficulties to control the robot in order to perfectly accomplish one specific task. The execution of complex tasks can be made impossible without having an estimated almost precise robot localization. A robust control system must have the capacity of self-localization in an environment, using the sensorial information and also the available information representing the environment (map) to estimate the robot position. This problem is also complex, and usually represents the “essential base” of a robotic control system.

The main goal of this work is to develop a robust control system for autonomous mobile robots that are able to operate and automatically adapt their behavior accordingly to different environments conditions. The control system must be able to determine the robot position using sensorial information and available environment map, even if the data are approximate and imprecise. The system must be able to keep, during navigation and tasks execution, an estimation of the correct robot position, starting with an initial approximate known position (local localization), or yet without any information about the robot initial position (global localization). During navigation the system should be able to detect and recover from incorrect position estimations (robot positioning error detection and relocalization). The system must be able to control the robot and navigate in a dynamic environment, preventing collisions with static and mobile obstacles. In order to achieve this goal a hybrid architecture for robot control and navigation was proposed and implemented.

In the next sections we will describe the proposed architecture – HyCar/Cohbra (section 2), the implemented simulator – SimRob3D – that was used to validate and to test our hybrid architecture (section 3), and some important results demonstrating the robustness and performance of our system (section 4).

2. HYCAR / COHBRA – Robot Control Architecture

In this work we present a new hybrid architecture of robot control - HyCAR/COHBRA (Hybrid Control of Autonomous Robots, or using the Portuguese acronym from “COntrole HíBrido de Robôs Autônomos”) [10]. This architecture integrates several important modules and components (data structures and methods) currently used to control autonomous mobile robots in one single block.

The mobile robot control architecture was structured in 3 layers: vital layer, functional layer and deliberative layer (Fig. 1). Each one of them is responsible for the reactive control (short term action), task execution control (mid term actions), and task planning (long

term actions), respectively. This type of system was also adopted in other similar works, like those developed by Gat [2] with the ATLANTIS architecture, and by Bonasso et al. [3] with the 3T architecture. However, our architecture proposes some extensions like the inclusion of the localizer module and the integration of multiple views of the environment (maps).

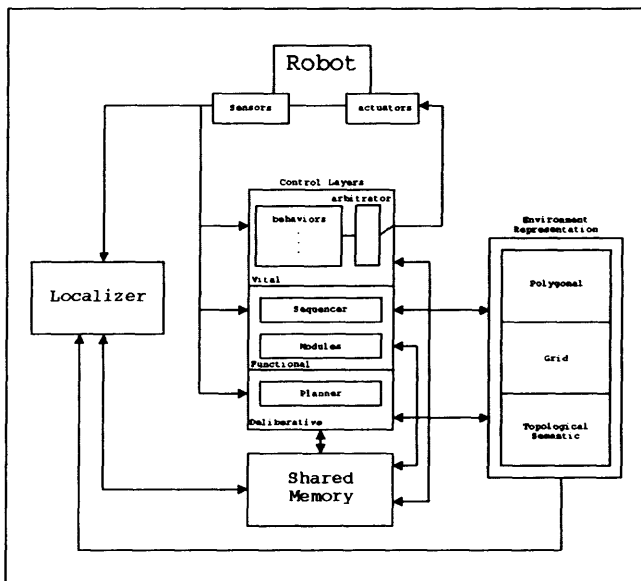


Figure 1 - Robot control architecture diagram, showing its main components: Control Layers (vital, functional and planner); Localizer; Environment Representation (polygonal, grid and topological maps); and Shared Memory.

In order to obtain a solid base for the execution of tasks implemented by the three Control Layers (vital, functional and planner), an additional module was integrated into the architecture: the localizer module. This localizer module must supply an estimated robot position related to the available environment map, and also using the sensorial data, it can continuously validate the estimated robot position. The localizer module is an essential part of our autonomous mobile robot control architecture, since we need an approximately correct robot position to perform tasks like: 'Move the robot from Office H to Room O'. If the system doesn't know where the robot is, it will be very hard to control it in order to achieve missions such as the above described task.

So, in the proposed architecture presented in this work, the localizer plays a central and essential role. The localizer allows us to determine the robot position, to detect changes in the environment maps and to detect static and dynamic obstacles. Since we can estimate the sensorial data according to a determined robot position and known environment configuration, then we can check it against the actual sensor data, and use this information to perform the above tasks (re-localize, detect changes and detect obstacles).

The specific form as the environment is represented internally in a control system, determines its precision and performance. Each one of the main known approaches applied to autonomous mobile robot control uses the most suitable environment representation adapted to some specific algorithm and/or purpose, choosing mainly between techniques like: discrete grids, geometrical maps, topological maps, probabilistic maps, Voronoy diagrams or potential field [11,12]. The proposed architecture we adopt in this work integrates

some of the most important approaches of environment representation, composing a hybrid scheme of environment maps, divided into layers: polygonal layer, grid layer and topological/semantic layer (Fig. 1).

The communication between all system components of our control architecture is allowed through a common data integration area implemented by shared memory. Through the use of this central repository of information, the several components of our architecture can exchange vital information for the perfect functioning of the autonomous mobile robot control system.

2.1 HYCAR / COHBRA Control System

Based on the concepts proposed in the HyCAR/COHBRA architecture [10] was implemented one specific control system. A special attention was given to the localizer module in the development of this system, as it is considered the main component of our control system and one of the main focus of this work. If the autonomous mobile robot possesses a good estimation of its actual position in the environment, then the navigation task becomes simpler and more precise. In the next sections we will describe the main components of the HyCAR hybrid control system: environment representation, localizer module and control layers.

2.1.1. Internal Representation of the Environment

In the internal representation of the environment we used together the three environment representation approaches defined in the control architecture: polygonal map, grid map and topological/semantic map. The polygonal map is mainly used by the localizer module to estimate the robot position, and it is initially supplied by the user in DXF format (AutoCAD Data) defining a blueprint of the environment (complete or approximate). The grid map is used by the deliberative layer to plan trajectories from a starting point to a goal destination (A* Algorithm [8]). The grid maps are generated from the polygonal map splitting the environment into cells using a predefined target resolution. The main function of topological/semantic maps is to improve planning performance, given some additional information that helps the planner to reduce space-state search and optimize the robot trajectory calculation.

2.1.2. Localizer Module

The Localizer Module was implemented using a Monte Carlo Localization (MCL) technique based on Fox et al [4] work. Monte Carlo localization approach possesses several advantages: it requires less computational resources than the majority of other techniques; it concentrates the resources and computational efforts in the areas of bigger interest for the localization; and as it is a probabilistic technique it can aggregate additional information about the robot position (e.g. certainty values associated to each estimated robot position).

One of the main reasons of our choice of a Monte Carlo localization method implementation and its integration into the HyCAR control system was its capacity to solve the three great problems related to robot localization: local localization, global localization, and re-localization.

In the local localization, the Monte Carlo localizer is able to keep a correct robot position from an initial specified position with an acceptable level of error. This method performs well during continuous robot displacements without the necessity of an external calibration function. The local localization is usually done when we start to use a mobile

robot, where the user assigns the approximate position of the robot in the environment (e.g. Office H on the left of the desk). The Monte Carlo localizer is also capable to perform a global localization, without need of any initial robot position information in order to determine the actual robot position. This localization technique is also capable of re-localize the robot, first detecting when the estimated robot position, apparently correct, does not reflect the actual robot position, and then performing a global localization to search for a new approximately correct robot position.

Monte Carlo Algorithm

The Monte Carlo localization (MCL) algorithm is divided in two phases: one phase of movement and the other phase of sensorial reading. Initially ' N ' samples (particles) are generated, uniformly distributed in the entire environment map (global localization), or distributed around the most likely robot position (local localization). Each sample is composed of a robot estimated position (x , y , direction) and the associated certainty.

In the *movement phase*, the robot is activated and the MCL algorithm generates new ' N ' samples, used to approximate the new robot position after this action. Each sample is randomly generated from the previous samples set. Samples are chosen from the old samples set with basis on their certainty that affects the probability of being preserved. Suppose L' being one position in the old sample. The new position L in the new sample is generated using $P(L | L', a)$, thus considering the observed action of movement ' a '. Before the correct robot position be achieved, it is possible to apply a certain threshold specifying the generation of some really new samples (samples renewal threshold).

In the *sensorial reading phase*, information are incorporated readapting the weight (certainty) of each sample in the set, using the sensorial values to estimate the certainty of each specific sample related to its position. This phase is mainly based on the idea that we can estimate the sensorial reading of the robot for a specific position related to the known environment map, and then compare this estimated sensorial data within the actual sensorial data obtained from the robot. From this comparison we can obtain the certainty value related to an estimated robot position.

To determine if the robot is or not correctly localized, we use the certainty value of the best sample (the one with the biggest certainty into the set) plus a dispersion measure of the whole set (that indicates if the particles are mainly concentrated around one specific position). These two information can help us to determine if the robot position was correctly estimated.

One problem of the Monte Carlo localization is that it is assumed that: (i) the environment is static; (ii) the representation of the environment corresponds to the real environment; and (iii) mobile obstacles do not exist. This kind of situation is not usually found in a real environment, for this reason it was necessary to use certain special techniques to treat this problem in our control system.

One approach to solve this problem was proposed by Fox [6] using "filters" to ignore certain readings from the sensors when these do not match with the expected inputs. Fox considered two techniques, entropy filters and filters of distance. The technique we adopted in our implementation of MCL module was the distance filter. This filter was better suitable to the kind of sensors implemented in our system (distance sensors).

Besides allowing the autonomous mobile robot to localize itself in a dynamic environment, the distance filter helps other modules of HyCAR control system. It makes available in the shared memory information about the filtered sensors, thus allowing the system to detect new static obstacles not present in environment maps. Filtered sensors indicate a difference between the internal representation and the real environment. The available data about filtered sensors is used by the functional layer to update the internal representation of

the environment. The only restriction of our system is during the localization process when we can work into a partially known environment but with no dynamic (mobile) obstacles. Once the robot position is known (correctly localized) then we can work in an environment with no restrictions: including unmapped static and mobile obstacles.

2.1.3. Vital Layer

The **vital layer** is responsible for reactive control of the autonomous mobile robot, composed of several simple process executed in parallel: elementary robot behaviors. These behaviors made an association between sensorial entrances and commands send to the actuators. Each behavior can be seen as a "sensorial-motor reaction", reacting directly to the environment stimuli.

We implemented 5 elementary behaviors in the vital layer, targeting to enable the robot: to follow trajectories calculated by the deliberative layer; to help the localizer module to determine the robot position; and to preserve the physical integrity of the robot. When following a pre-defined trajectory the autonomous mobile robot must be able to avoid and deviate from the unexpected obstacles (static or dynamic). The five behaviors implemented in the vital layer are: (i) stop; (ii) wander; (iii) avoid obstacles; (iv) move in direction to near target position; and (v) go back. The "avoid obstacle" behavior is based on the potential field method (VFF) as used by Borenstein & Koren [5].

The interactions between these behaviors are managed through an *arbitrator*. The arbitrator has the function of activate or inhibit certain behaviors, depending on the commands received from the sequencer in the functional layer. The arbitrator can also be programmed to compute fusion rules of behaviors outputs, in order to integrate ambiguous outputs from different behaviors.

2.1.4. Functional Layer

The **functional layer** is composed of various modules that interact among themselves, executing different functions responsible for the integration of the control system components. One of these functions of this layer is to select which elementary behaviors from the vital layer will be executed at a time, and to provide parameters to these elementary behaviors (e.g. next target position to move). With the information provided by functional layer and using the specification of the execution sequence of these elementary behaviors, the robot can execute high level tasks defined by the deliberative layer planner.

The *sequencer* of the functional layer was implemented in the form of a finite-state automaton. Each state of this automaton indicates for the *arbitrator*, in the vital layer, which behaviors must be set on or must be inhibited. So, the sequencing is executed through inhibitions of some specified outputs coming from elementary behaviors. There is no module in the functional layer that can act directly in the control of the robot actuators, which are controlled by the arbitrator in the vital layer. The modules of the functional layer are responsible for supplying information to the elementary behaviors of the vital layer that will be active, and inhibit the outputs of the behaviors that should be deactivated.

The functional layer is also composed by other functional modules that play auxiliary tasks in the control process of the autonomous mobile robot. In the HyCAR control system where specified a series of auxiliary functional modules. One of these modules is responsible for monitoring and update the internal representation of the environment, and the others provide parameters that assist other modules in different control layers.

2.1.5. *Deliberative Layer*

The **deliberative layer** has the only function to perform trajectories planning, from the robot position to a final user defined position. The planner may be activated by the user (new destination position specification) or it can be interrupted by requests originated from the functional layer. Sometimes the functional layer can detect an impossibility of plan execution (route no longer available due to obstacles) and then request a new planning.

The planning task is processed in two phases. In the first one, using the topological information (connectivity graph), a pre-planning is executed and determines the sequence of topological regions that will be traversed in the final path. The Dijkstra algorithm [7] was used in this initial phase, and this information will be used to optimize the next phase of planning. In the second phase, the A* algorithm [8] is used to calculate the definitive trajectory based on a grid representation of the environment map. The grid representation based algorithms can be great CPU time-consuming tasks, depending on the grid size. In our system, this task is optimized using the pre-planning phase to reduce the size of the grid used by the path-planning algorithm.

The trajectory planning obtained using the A* algorithm produces a way composed by a sequence of cells that must be followed to reach the destination position. This sequence of cells is converted into a sequence of points into the polygonal representation using the center of cell as base. The final plan is one trajectory to be followed by the autonomous mobile robot, from its current position to the destination position, composed by a sequence of points in the polygonal layer representation. This plan is made available into the shared memory to be used by the functional layer.

3. SimRob3D Simulator

We implemented a robot simulator that includes all necessary resources to realize experiments with autonomous mobile robots placed in a dynamic environment. This implementation was created in order to validate our proposed control system. This simulator was called SimRob3D (Mobile Robots Simulator based on Three-dimensional Environments) [10]. The main characteristic of this simulator resides in the fact that it implements our proposed control architecture using a three-dimensional environment for the navigation of the simulated mobile robots. The environment structure and objects can be designed with three-dimensional modeling software existing in the market (AutoCad, 3D Studio, among others), once we adopted in our simulator a common standard 3D data file format, the ".3DS". This file format allows us to specify different elements composing the robot environment (walls, objects, light, textures), resulting in an environment with a high level of realism if compared with other environments used in some 2D based simulators.

The simulator allows the user to place and configure obstacles, also allowing moving them in real time during simulation. The obstacles can also be pre-programmed to move in cyclical trajectories. The simulator possess several sensorial and kinematics models (e.g. encoder, sonar, infrared and laser sensors; Ackerman and differential kinematics), allowing the user to configure different types of robots.

All the sensors and actuators interact directly with the three-dimensional environment, becoming the simulation more realistic. Sensor and actuators are also modeled in a realistic way, i.e. we included in their model a gaussian error in order to introduce input sensor errors (e.g. noise) and output control errors (e.g. wheels sliding). These models of sensors and actuators provided to us a more realistic and non-deterministic robot behavior.

Another important characteristic of our simulator is its modularity. The controller is programmed separately from other simulation functions, implemented as a dynamic library

(DLL). The controller is loaded at execution time, and it can be implemented using any language chosen by user. As the controller is completely separated from the robot and environment simulation, it has a well defined interface to communicate with the robot. The only information exchanged between robot controller and the other simulation modules are basically “*Get_Sensors*” and “*Send_Command*”. So, we can easily replace the simulated robot by an actual robot. HyCAR controller was implemented in that way, as a separated module that communicates with other SimRob3d modules.

4. Simulation Results

We used SimRob3D to validate HyCAR robot control architecture in *localization* and *navigation* tasks. Several *localization experiments* in static and dynamic environments were carried out: using a perfect environment map and using a modified environment map (with some static and dynamic/mobile obstacles added, which are not present in the original map). We evaluated the capacity of our system to perform local and global localization, and also we test the ability of the system to re-localize the robot when an incorrect initial position was informed to the robot. These experiments were executed using the original Trinity environment map [9] (figure 2). We also used some variations of that environment, in order to evaluate each type of localization, where the robot available environment map was set to the original Trinity environment, but during simulation the robot interacted with a modified map with some removed objects and added obstacles.

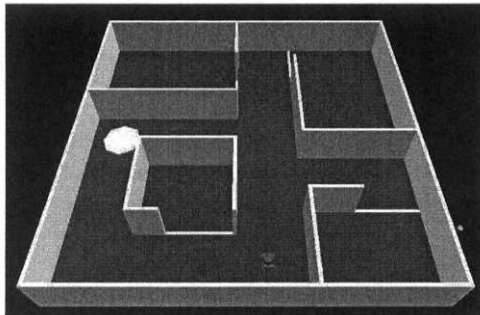


Figure 2 - Three-dimensional environment used in localization and navigation experiments based upon Trinity Fire-Fighting Contest environment map [9].

Table 1 shows the simulation results, using a 6x8cm Ackerman based robot, with the localizer module configured to use: 1000 particles sampling, 16 sonar sensors (30 degree wide range field, 100.0 to 1.0 cm distance sensibility, 5% of average noise) and 2 actuators - wheels and steering (10% average error on encoders, 1% average error on command response). Experiments were made with robot following a pre-defined trajectory, with a total of 2700 simulation cycles. Each cycle corresponds to a set of sensorial readings received from the robot and used by the localizer module to estimate the robot position. Table 1 values are the average obtained from 10 different simulations results.

Table 1 - Results obtained in localization experiments.

Experiment	Number of Localization Cycles (Correct Position Found)	Number of Localization Cycles
Local Localization	0	2700
Global Localization	674.4	2025.5
Re-localization	681.8	2018.2

Navigation experiments were carried out in static and dynamic environments with some static or mobile obstacles added on it. The main objective was to evaluate the capacity of the control system to conduct the robot in these environments, following the plans supplied by the deliberative layer. The robot was placed in an initial position and the user specified several target destinations.

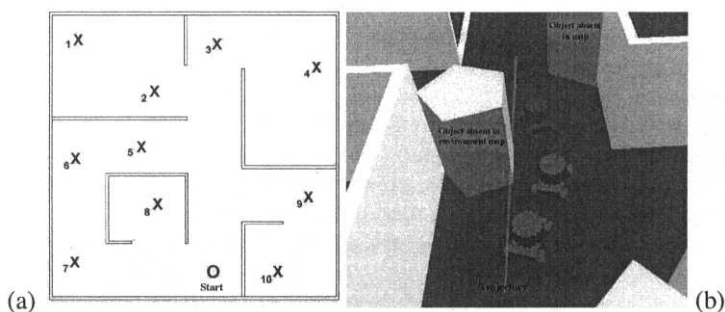


Figure 3 - Trinity environment map with destination points used in navigation experiments (a). Robot movement sequence executed to avoid an unexpected obstacle (b).

The results obtained in navigation experiments had shown that, in all simulations (10 simulations for each experiment: from the initial position, move to destination points 1-10; see Fig. 3(a), the control system was able to guide the mobile robot and achieve destination position, deviating from all unmapped obstacles (static or mobile), as can be seen in Fig. 3(b). The localizer module was able to keep a correct robot position during the entire trajectory, with an average certainty of 98% and a dispersion of 0.73cm with 91% particles grouped. Calculating the difference between the controller estimated position and the simulated robot position (actual position), we obtained a positioning error of 2.16cm in average computed over all simulation execution, in a 6.15m^2 environment.

We also carried out some experiments to determine if the use of a hybrid environment map representation can improve the planning algorithm execution. We used the same navigation tasks as described in Fig. 3(a). The planning execution time was measured for A* algorithm with no pre-planning and after that using our pre-planning algorithm based on topological information. The obtained results showed an improvement of global planning time execution with average gain of 68.5%.

These results demonstrate an efficient integration between the deliberative layer and the vital layer (reactive). The robot was capable to accomplish different navigation tasks even in presence of a modified environment with mobile obstacles. The localizer module also demonstrated the capacity to keep a correct robot position estimation even in a partially known environment (modified environment).

Our experiments demonstrated that our hybrid control system was able to keep an estimated correct position in all simulations configurations, with very good local localization ability. The experiments with global localization and re-localization obtained good results in most of tested environments, having some difficulties only in environments with a great number of modifications. In the navigation tasks, the control system was perfectly able to move the robot up to the specified objectives in all types of environment we used in our experiments. Some videos showing SimRob3D experiments can be viewed in the simulator site on the Internet: http://necg.unisinos.br/robotica/simulador/index_us.html

5. Conclusion

The results had shown that HyCAR hybrid control system was perfectly able to control autonomous mobile robots. The system was able to execute navigation tasks in static and dynamic (modified) environments also including the presence of mobile obstacles. The mobile robot was able to follow global trajectories planned by the deliberative layer module, and using their reactive skills from the vital layer module, it was possible to navigate without colliding with any kind of obstacle or being imprisoned by local minima, even in a dynamic environment. Our experimental results showed that cooperation between the global planning algorithm and the local reactive algorithm (intermediated by the functional layer) provided a robust robot control and navigation system. The localization module also shows a capacity to determine and to keep a good estimation of the robot position in the environment.

The proposed architecture fulfills our main goals: (i) provides a global navigation system with a low computational cost; (ii) provides a robust local navigation system, preventing the robot to stay blocked in a local minima or to collide with unmapped obstacles. The localization module has a central role in this system, since this module provided indispensable conditions to correctly control the robot in order to achieve the proposed tasks. This demonstrates that it was possible through the combination of different methods to obtain a robust control system integrating and exploiting the best characteristics of each one of these methods.

The main contribution of this work was the proposal of a new hybrid control architecture for autonomous mobile robots that was validated through experiments. The HyCAR system proved to be robust and capable of operate in dynamic environments. The HyCAR architecture is well adapted to operate an autonomous mobile robot in changing environments that can include mobile obstacles, and also it is able to determine the robot position in these environments even when the environment did not reflect exactly the internal representation map. In the near future we are planning to integrate our control system into a Nomad 200 robot and then evaluate our control system in real situations.

References

- [1] G. Dudek and M. Jenkin. **Computational Principles of Mobile Robotics**. Cambridge University Press, Cambridge, UK. 2000.
- [2] E. Gat. **Integrating Planning and Reacting in a Heterogeneous Asynchronous Architecture for Controlling Real-World Mobile Robots**. AAAI-92 Proceedings, AAAI Press, 1992.
- [3] R. P. Bonasso., et al. **Experiences with an Architecture for Intelligent Reactive Agents**. Journal of Experimental and Theoretical AI, 9(2). 1997.
- [4] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. **Monte Carlo localization: Efficient position estimation for mobile robots**. In Proc. of the National Conference on Artificial Intelligence (AAAI). 1999.
- [5] Borenstein, J. and Koren, Y. **Real-time Obstacle Avoidance for Fast Mobile Robots**. IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19, No. 5, pp. 1179-1187. 1989.
- [6] D. Fox. **Markov Localization: A Probabilistic Framework for Mobile Robot Localization and Navigation**. Institute of Computer Science III, University of Bonn, Germany. Doctoral Thesis. 1998.
- [7] T. Cormen, C. Leiserson, R. Rivest. **Introduction to Algorithms**. MIT Electrical Engineering and Computer Science Series. MIT Press. 1990.
- [8] Nilsson N. J. **Principles of Artificial Intelligence**. Tioga Publishing Company. 1980.
- [9] Trinity College. **Fire-Fighting Home Robot Contest Website**. Ultima Atualização: Abr 2002. "<http://www.trincoll.edu/events/robot/>".
- [10] Heinen, Farlei. **Sistema de Controle Híbrido para Robôs Móveis Autônomos**. Master Thesis. UNISINOS University, PIPCA-Applied Computing Master Course. São Leopoldo-RS, Brazil. May 2002.
- [11] Dudek, Gregory and Jenkin Mitchel. **Computational Principles of Mobile Robots**. Cambridge University Press. 2000.
- [12] Murphy, Robin R. **An Introduction to AI Robotics**. MIT Press. 2000.

This page intentionally left blank

Section 14

Web and Multimedia Applications

This page intentionally left blank

Clustering Web User Interests Using Self Organising Maps

Xiaozhe Wang and Kate A. Smith

*School of Business Systems,
Faculty of Information Technology,
Monash University, Clayton, Victoria 3800, Australia*

Abstract. This paper presents an approach to clustering a Web user's interests represented as text term maps using the unsupervised Neural Networks algorithm (Self Organising Map) [6] from the records in the particular user's history file. Self Organising Map is a good tool for clustering the text data set into a low-dimensional regular grid that can be visualised as maps labeled with text terms. In this research, an experiment was carried out to find the User Interests Term Map in which all associated terms could be grouped in the same cluster. The text terms based on the Web user's interests could be applied to the intelligent Web query search in future work.

1. Introduction

With the growing popularity of the World Wide Web (WWW) and the appearance of new technologies on the Internet, intelligent Information Retrieval (IR) has become increasingly important for Web users, and especially non-professional users. An intelligent search mechanism with flexibility and adaptability can help users focus on their own preferences and interests to improve the precision of IR on the Web. A Web personalisation system could perform the task for each individual user by finding the user's interests automatically.

The dynamic features of the Internet environment can lead to low precision and low effectiveness of IR if conventional search methods are [12]. When users with different backgrounds or interests search for different information based on the same query term retrieval method, the result cannot satisfy the purpose of "real understanding of what the user means" with the traditional IR process using current search engines. For example, in the traditional search system, if the user inputs a query with several keywords, the system returns the exactly same result to all users who input the same query. Therefore, the search result is not personalised or adaptive to the user's particular needs. To find the exact information the user needs based on the limitation of the terms match algorithm used by the conventional search and indexing systems, they have to modify the query terms in an "ad hoc" or trial and error manner. While, professional users try to modify their query input to get the specific information they require. However, non-professional or inexperienced users may not know how to do this. To overcome this problem, it becomes necessary to create a user profile of interests for each individual user. This can be achieved by training the search system to learn the interests of an individual user automatically and continuously. Identifying the interests of the users helps to modify the query automatically based on the user's interests and increases the effectiveness of the IR system for Internet searching. In recent research on automatic personalisation systems, the "user profiling" has become a common approach in performing the task of intelligent Web IR and Web searching.

In this paper, a novel approach is proposed for finding the interests of a Web user by clustering the text terms contained in the URL titles stored in the user's history file on the

Web browser. The user's interests are organised and presented as different clusters with corresponding text terms.

The history file is chosen as the training data for clustering the user interests terms. Web mining consists of content mining and usage mining. In the former, the content of the Web pages is the objects being analysed to discover content patterns of Web site. As in some news group applications, the user profile is constructed with the terms represented as a vector model or database which extracts information from the documents read by the users and the feedback text documents. Similarity in the bookmark service from the browser, the Web pages referred to by each of the URLs are used as the documents to extract the terms for generating the basic data used for Web content mining.

The proposed clustering approach is for performing Web content mining, as the user's interests to be identified are represented as text terms extracted from the user's history file. Using bookmarks and Web pages as documents to extract the training data has some disadvantages. First, both methods are extremely time-consuming for processing the large data sets used in the system. Second, both methods need highly explicit user participation. Without the co-operation of the user, there would not be sufficient documents generated to do the clustering and training required for finding their interests, as the bookmarks and Web page feedback have to be created manually. However, if the history file is used, these problems are overcome by the automatic information storage function for each of the URLs visited on the Web browser. The history file contains information such as the time frame was first and last visited and the number of similar visits, analysis of which can make the data about users interests more precise. As such, the history file is suitable for finding the hidden relationships between all the text terms to represent the user's real interests.

This approach is based on Kohonen's Self Organising Map (SOM) which is an unsupervised Neural Networks (NN) learning algorithm [6]. SOM can accept as input objects described by their features and place them on a two-dimensional (2D) map in such a way that similar objects are placed close together. In this sense, it is similar to multidimensional scaling [5]. Because it is hard to know and control the preferences of each Web user, the SOM's unsupervised NN technique is suitable for performing the training and clustering with limited available data to predict useful "user interests". Another feature is its high capability to deal with large data sets, which is also suitable and highly required by Web applications. Different users have different browsing habits on the Web.

Normally, the contents and records in the history file are a very large data set with continuous growth of the information browsed by the user on the Internet. As such, SOM for analysing and visualising document collections is an appropriate approach for IR and data mining [4]. Based on SOM's potential ability of overviewing the content of large document collections, association searches and query extension become easy to use as a personalised search method. The explicit word match system can be replaced by a user interest-oriented system for a more desirable result.

Compared with previous research on "user profiling", this approach overlaps the gap between Web content mining and Web usage mining. Normally, the approaches within either direction focus on content or usage information only. However, in order to generate the text term clusters based on usage patterns, the SOM algorithm is applied in this approach to perform the task for finding the content-oriented clustering with user's natural browsing habit as input features.

The following section provides an overview of the current research on user profiling in IR system. Then Section 3 describes a novel clustering approach based on SOM for generating the cluster map of the user's interests. To demonstrate the effectiveness of the approach, an experiment is conducted with a sample data set using Viscovery SOMine [16]

in section 4. In the last section, some concluding observations are made and future work proposed.

2. User Profiling in Information Retrieval

The common approach to achieve Web personalisation is to create user profiles using different algorithms and techniques.

Information agents are used to extract the user profile and interests from his/her bookmark records [7]. Compared with the current search engine, this approach has some certain advantages. Using the bookmark is not ideal due to its heavy dependency of the users' participation (as mentioned in Section 1). Content-based filtering largely depends on the information provided by the user, so it cannot provide the precise result to the user if less information is provided [3]. If the bookmark is used as the data source to extract the features of the user's interests, some sophisticated techniques for automatically updating and organising of the bookmark have to be used in the system such as "Powerbookmarks" [8],[14].

Many information-filtering systems based on the Vector Space Model (VSM) are using simple measures of word frequency [14]. As such, the user profiles are set up as a VSM based on the relevance feedback method from the formula TF (Term Frequency)-TFDF (Term Frequency and Document Frequency [12],[2]. But, this method has potential problems for online systems due to involving the large text data concerned processing speed limitation. The VSM user profiling method was upgraded further into multiple interest vectors whose number, size, and elements change adaptively based on user access behavior as Multi-Modal (MM) [1].

The NN model is another means of creating the user profile, such as Adaptive Resonance Associative Map (ARAM) [15]. ARAM represents a user's interest profile by a set of recognition categories, each associating a set of conjunctive features to a relevance factor. In other applications, user profiles are created by using the Fuzzy Logic (FL) techniques [9]. In comparison with the NN techniques, FL techniques can make the output of the user profile more meaningful but with a disadvantage is that it requires specified decision rules to do the training and learning process.

3. The Clustering Approach Using SOM Algorithm

The approach proposed in this paper is to infer and reveal the user interest patterns by clustering the content of the history file using the SOM algorithm. As such, the hidden irrelevant records in the history documents can be effectively organised into the clusters for automatically personalising the Web search. The Figure 1 shows the process.

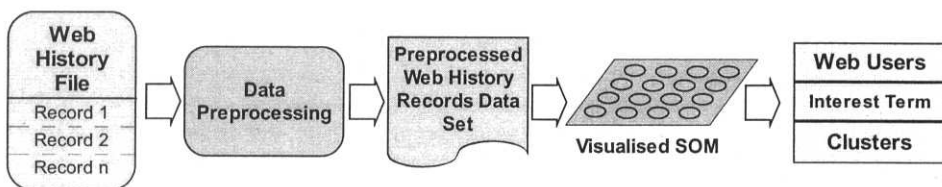


Figure 1: Generating User Interest Clusters using SOM

3.1 Data Preparation

There are several preprocessing tasks to perform before the SOM algorithm can be applied to the data set from the user’s original history file. It is very important in determining final result, because different inputs can generate a totally different result with the same technique applied.

Data preprocessing involves data formatting, cleaning, classifying and representing in this approach, outlined as follows.

3.1.1 Data Formatting

The history file from the Web browser can be retrieved and saved into different formats with different purposes. Normally, the history file can be represented as a text file or HTML file regardless of the browser used. Table 1 below shows a sample text file representing the saved history file from the Netscape browser:

Table 1: Sample Netscape History File Saved in Simple Text Format

<!DOCTYPE NETSCAPE-history-file-1>
<DT><AHREF="http://www.infotech.monash.edu.au/"FIRST_VISIT="1012878212" LAST_VISIT="1024143821" VISIT_COUNT="512">Information Technology - Survey
<DT><AHREF="http://www2.ems.uq.edu.au/phdweb/phhome.html" FIRST_VISIT="1024103884" LAST_VISIT="1024103884" VISIT_COUNT="1">PhD - First thoughts to finished writing

Each history record contains (a) the detail URL of the each visited page (b) “First visited time” and “Last visited time” for each URL (c) “Visited Total times” and (d) “page title” representing the main content of each visited page.

In order to proceed with the data preprocessing, all the original data from the history file is converted into a spreadsheet format as shown in Table 2.

Table 2: Formatted Sample Data from the Simple Text History File

URL	FIRST_VISIT	LAST_VISIT	VISIT_COUNT	Title Terms
www.infotech.monash.edu.au	1012878212	1024143821	512	Information Technology Survey
www2.ems.uq.edu.au/phdweb/phhome.html	1024103884	1024103884	1	PhD First Thought Finish Writing

3.1.2 Data Cleaning

Depending on the preferred setting of the Internet browser, the user can decide the time frame that the record will be kept in the history file. In order to make the data set meaningful for the process, data cleaning has to be completed before the implementation of the SOM algorithm. Generally, users visit Web pages and Web sites relevant to their interests or preferences. For example, if a user visits a Web page many times, it suggests a particular interests in the information of the Web page or that is a useful information source frequently needed.

By specifying a threshold value, T , for the “Total Visit Times”, the records in the original history file can be first cleaned. For example $T = 1$, then all the URLs that are visited only once will be removed from the data set. Noisy data in the history record to be cleaned as well because these cannot reflect the real interests for the purpose of generating the “user profile”, including the following:

- Email Service records
- Failed searching records
- Basic search functions like “forward” or “back” in the history file
- Internet common terms, such as “WWW”, “net” and so on
- Interests irrelevant words, such as “to”, “in” and so on

3.1.3 Data Classification

After the data set is cleaned by a filtering system based on a comparison with the threshold T and other cleaning rules, it still needs to be classified into sub-sets for training it is very large. As with any techniques for the computing process, if the data set is very large, it not only consumes large amounts of processing time but also limits the applicability of the system to data in the real world [10]. Because many thousands of records are kept and each record contains multiple text terms which are used as the basic object input for the clustering process, data from the history file must be classified. As such, specifying a particular Time Frame $TF_n \in (D_{\min}, D_{\max})$, which covers the time period from D_{\min} (most recent day) to D_{\max} (farthest day) for classifying the data is essential. As mentioned previously, the maximum days of keeping visited pages in the history file can be different from user to user due to the different setting in the browser. Whatever the setting is, a more recent time can represent a higher degree of the interest of the user. However, in the real application based on the different Web users, for the history file which contains large amount of visited records, it can then be classified into n small sub data sets by the time frame TF_n in which the number n can be set as the parameter according to the requirement and the quantity of the records in the user’s history file. The basic rule is that the weight of the cluster result decreases when the value of n increase. In the experiments carried based on our approach, the number n is 1 for the sample data set, that means the whole data in the history file is treated as one data set.

3.1.4 Data Representation

The input objects in the proposed approach are text terms. The representation of the text terms is crucial before the terms can be organised by SOM from the computational point of

view. All the text terms inputs have to be translated into a suitable numerical form for processing in the SOM algorithm. A numerical value assigned to each text term alphabetically sorted and is used as Term_Code for text terms' representation.

Other inputs for each text term include the Visit_Count_Frequency, the Day_since_Last and the Day_since_First as shown in table 3 below. The frequency count represents the textual content term with a numerical value is a common approach used to represent the textual content terms in IR. In this approach, the frequency for each text term is represented by using the visited times count record in the history file. The scaling method is used for representing the text terms' random coding system of the records and the Day_since_Last and Day_since_First data are converted from the history original data by calculated with maximum and minimum value of the records in the history file. Table 3 shows the features as inputs to the SOM algorithm.

Table 3: Sample Data Representation for SOM Algorithm

Term	Term_Code	Day_since_First	Day_since_Last	Visit_Count_Frequency
Artificial	0.02	22	2	3
Australian	0.04	27	7	6
Bear	0.06	18	8	5
Book	0.08	29	9	6

3.2 Constructing the User Interests Term Cluster Map

Most conventional text mining algorithms aim at finding text terms based on the term similarity. In this approach, the text terms are classified based on the user's interests from browsing Web pages hidden in the title records of the history file. The terms belonging to the same cluster do not necessarily have the similar meaning. They appear in a similar time period with a similar occurrence of the times counted in the user's history file that is the main feature for finding the user's interests for Web browsing. The input text terms are multi-dimensional data before being analysed by the SOM [6] algorithm. After the unsupervised training process with the SOM algorithm, a 2D map space is generated to present the term nodes and the relationship between the clusters with easy visualisation. This result can be applied to the Internet application directly for a visual illustration of the particular user's interests. By processing the history file with the SOM algorithm, the user's interests can be identified with less, yet still sufficient, data more efficiently and effectively.

A 2D map of the user's interest term clusters is formed after the training process with the unsupervised learning SOM algorithm. The interests-related text terms are grouped into the same cluster on the map. At the same time, the relationship between different clusters can be explicitly shown on the map.

4. The Experiment with the Sample Data Set

In the experiment, data source came from one single user's history file of Netscape Web browser with 60 days time period. The original history file contains 562 entries of visited URL's in the particular historical time frame. For each entry, there are normally consisting more than 3 text terms. After preprocessing, all the encoded text terms with 4 extracted

feature inputs, were used for training to generate the map using Viscosity SOMine. 6 interest text term clusters were obtained from the training process with setting of 1000-node and 0.3-tension. Each node represents a part of the source data set and tension is the parameter characterizes the influence radius of the neighborhood interaction in the map during the final stage of the training process. The cluster result was finalised when “normalized distortion” which measures the fitting of the map with respect to the shape of the data distribution was 0 and the “quantization error” which measures the average square distance of all records in the data set from their respective projections onto the map was 0. This was normalised with respect to the current tension of the map.

The computer system used in the experiment was Microsoft Window 2000, Intel®Pentium 4 CPU 1.50GHz, AT/AT Compatible, 261,616 KB RAM. In order to obtain the best result from the limited input, different parameter settings were used in the experiments. During the training process the nodes of the map gradually adapted themselves to the intrinsic shape of the data distribution.

The training process's speed was different with different settings of the number of nodes. With a large number of nodes setting, the process consumed time longer. For example, if given same tension setting 0.3, but with different settings for number of nodes as 500, 1000 and 2000, the consumed time for training process were 4, 16 and 49 seconds respectively. That is, the lower the number of nodes, the faster the converging process for the whole process.

However, the results of the clusters were not satisfactory with too few node settings. In the experiment, a 500-node setting generated 16 clusters, but above 1000-node settings gave the same result for number of 6 clusters even though the detail text terms were different within each cluster.

In addition, different settings for the tension parameters also affected the result. A high-tension setting resulted in a map in which “averages” the data distribution, while smaller values allowed the map to adapt its shape to finer details.

Compared with the results from the different parameter settings, the best result obtained with the parameters setting of 1000-node and 0.3-tension because all errors were minimised as 0 for “normalised distortion” and “quantization error”. For ensure the experiment result was not conducted as incident, with same parameter setting, the experiments were performed and tested for several times and the promised cluster results obtained each time. Therefore, the training process was finally converged at the end.

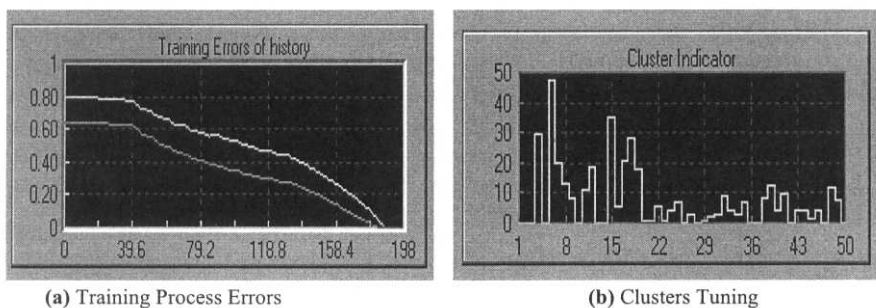


Figure 2: Training Process Graph and Clusters Tuning Graph

Training of the experiment from start to finish totalled 16 seconds with 976 nodes in the final map after 25 cycles of training. As in the above Figure 2 (a) training process errors graphs, error of “normalised distortion” as blue line decreased from 0.8 to 0 and

Table 4: Sample UITCM Cluster Text Terms from Experiment

Cluster 1 Terms	Cluster 2 Terms	Cluster 3 Terms
Business Commerce Computational Conference Consultation Finish First Google Information Intelligence International Internet Monash Name	Australian Bear Book Clothing Council Discovery Ethics Free Grant Pattern Research Sale Sewing Teddy Women	PHD Research Retrieval School Search Soft System Thought Time Tutors Writing
Cluster 4 Terms	Cluster 5 Terms	Cluster 6 Terms
Business Monash School Subject Survey System Technology Trading University	Book Cutting Grant Machine Monash Pattern Scheme Sewing	Agent Artificial Commerce Computatational Computing Conference Google International

5. Conclusion and Future Work

In IR system, there are two common ways [13] to extract the user's preferences. One is the "Direct Extraction" which asks users to register their interests in the form of keywords, topics, and so on. The other, "Indirect Extraction", records each visited documents page. Both methods create some problems: the former requires much effort on the part of the user and their real interests cannot be precisely specified when the users are uncertain of their needs or interests; with the later, it is necessary to use appropriate techniques to record the time log and page information, and to assume that the user reads the page without a break. The approach used in this paper combined the two methods with an automatic training process to achieve better result.

The user interests term cluster result can be used directly for intelligent Web search query updating and Web page ranking. The search query can be extended based on matching with other relevant interests terms within the same cluster of the original query term input. For example, when modifying the query is not applicable, the relevant terms in the same cluster can be used to supplement the user's original query to retrieve more precise result as what they need. Therefore, the text terms in the user interests term clusters can be used for refining, updating and extend the query for retrieving the information as adaptive Web search. Meantime, for search results with numbers of page links, the terms in

the same cluster also can be used to perform the ranking task to help single user find the information more effectively and efficiently.

Because the personalisation in Web mining is a new research direction, still no definite general methodology is available so far although many work done with existing systems for limited domain. It is necessary that IR systems tailor the retrieved document set as per users' history or nature [11]. The relationship between different clusters needs to be discussed further for clarifying the characterization or representation of each cluster. Meantime, how to make the clusters meaningful with limited capability of data source is still need to build up the appropriate techniques to verify as the future work.

Acknowledgements

First author would like to thank Dr. Chung-Hsing Yeh for the valuable comments that helped to improve the clarity of this paper.

References

- [1] Cetintemel U., Franklin M.J., and Giles C.L., "Self-Adaptive User Profiles for Large-Scale Data Delivery", In *Proceeding of the International Conference on Data Engineering*, 2000, San Diego, CA.
- [2] Chen, P.M. and Kuo F.C., "An Information Retrieval System Based on An User Profile", *The Journal of Systems and Software*, 2000, 54: pp. 3-8.
- [3] Harman, D., Braschler M., Hess M., Kluck M., Peters C., Schäuble P. and Sheridanet P., "Relevance Feedback and Other Query Modification Techniques", In *CLIR Evaluation at TREC. 1992, SIGIR Forum* 27(3). pp. 241-263.
- [4] Honkela, T., Lagus K. and Kaski S. and Kohonen T., "Self-Organizing Maps of Large Document Collections", *Visual Explorations in Finance with Self-Organizing Maps*, G. Deboeck and T. Kohonen (eds.), Springer, 1998, pp. 168-178.
- [5] Jain, A.K. and Dubes R.C., "Algorithms for Clustering Data", 1988, New York: Prentice-Hall.
- [6] Kohonen T., "Self-Organizing Maps", Berlin, 1995, New York: Springer.
- [7] Liu, B., Wang H. and Feng A., "Applying Information Agent in Open Bookmark Service", *Advances in Engineering Software*, 2001, 32: pp. 519-525.
- [8] Maarek Y.S and Shaul I.Z.B., "Automatically Organizing Bookmarks per Contents", *Computer Networks and ISDN Systems*, 1996, 28: pp. 1321-1333.
- [9] Martin-Bautista M.J. and Kraft D.H., "User Profiles and Fuzzy Logic in Web Retrieval", In *Proceeding of Joint Eurofuse-Sic '99 International Conference -Fourth Meeting Of The Euro Working Group On Fuzzy Sets And Second International Conference On Soft And Intelligent Computing*, 1999, Budapest, Hungary.
- [10] Ng A. and Smith K.A., "Web Page Clustering Using A Self-Organizing Map of User Navigation Patterns", In *Proceeding of Smart Engineering System Design: Neural Networks, Fuzzy, Logic, Evolutionary Programming, Data Mining, and Complex Systems*, 2000, Missouri, USA.
- [11] Pal S.K., Talwar V. and Mitra P., "Web Mining in Soft Computing Framework: Relevance, State of the Art and Future Directions", *IEEE Transactions on Neural Networks*, 2000.
- [12] Park Y.W. and Lee E.S., "A New Generation Method of An User Profile for Information Filtering on the Internet", In *Proceeding of the 13th International Conference on Information Networking*, 1998, Tokyo, JAPAN: the IEEE Computer Society.
- [13] Sakagami H. and Kamba T., "Learning Personal Preferences on Online Newspaper Articles from User Behaviors", *Computer Networks and ISDN Systems*, 1997, 29: pp. 1447-1455.
- [14] Salton G., "Automatic Text Processing: the Information, Analysis and Retrieval of Information by Computer", 1989, Addison-Wesley.
- [15] Tan A.H. and Teo C., "Learning User Profiles for Personalized Information Dissemination", In *Proceeding of International Joint Conference on Neural Networks*, 1998, Alaska.
- [16] Viscovery SOMine, <http://www.eudaptics.com/technology/somin4.html>.

Web Traffic Mining Using a Concurrent Neuro-Fuzzy Approach

Xiaozhe Wang, Ajith Abraham and Kate A. Smith

*School of Business Systems, Faculty of Information Technology,
Monash University, Clayton, Victoria 3800, Australia*

Abstract. Web servers play a crucial role to convey knowledge and information to the end users. With the popularity of the WWW, discovering the hidden information about the users and usage pattern is critical to determine effective marketing strategies and to optimise the server usage and accommodate future growth. Many of the currently available server analysis tools could provide only statistical data without much useful information. Mining useful information becomes a challenging task when the user traffic volume is enormous and keeps on growing. In this paper, we propose a concurrent neuro-fuzzy model to analyse useful information from the available statistical/text data from the Web log analyser. We made use of the cluster information generated by Self Organising Map (SOM) [7] for data analysis and a Fuzzy Inference System (FIS) to forecast the daily and hourly traffic volume. Empirical results clearly demonstrate that the proposed hybrid technique is efficient and could be extended to other Web environments.

1. Introduction and Motivation for Research

The World Wide Web (WWW) is continuously growing with rapid increase of the information transaction volume and number of requests from Web users around the world. For Web administrators and managers, discovering the hidden information about the users access or usage patterns has become a necessity to improve the quality of the Web information service performances. From the business point of view, knowledge obtained from the usage or access patterns of Web users could be applied directly for marketing and management of E-business, E-services, E-searching, E-education and so on.

However, the statistical data available from the normal Web log data files or even the information provided by Web trackers could only provide the information explicitly because of the nature and limitations of the methodology itself. Generally, one could say that the analysis relies on three general sets of information given a current focus of attention: (1) past usage patterns (2) degree of shared content and (3) inter-memory associative link structures [12]. Computational Web Intelligence (CWI) [18], a recently coined paradigm, is aimed to improve the quality of intelligence in the Web technology [11]. The pattern discovery of Web usage mining consists of several steps including statistical analysis, clustering, classification and so on [13][10]. Most of the existing research is focused on finding the patterns; with little efforts on the detailed pattern analysis. We propose SOM [7] to cluster and discover patterns from the data. These clustered data were further used for different statistical analysis. In order to make the analysis more intelligent we also used the clustered data to forecast the daily traffic volume and the hourly page requests. Using a Takagi Sugeno Fuzzy Inference System (TSFIS) [14], we explored the prediction of average daily traffic volume (1 to 5 days ahead) and the hourly page requests traffic in a day (1,12 and 24 hours ahead).

As a case study, we explored the Web user access patterns of Monash University's Web site located at <http://www.monash.edu.au>. We made use of the statistical data provided by 'Analog' [3] Web access log file analyser, which is embedded at the

University's Web server. 'Analog' generated data and text information covers different aspects of the users' access log records, weekly-based reports include traffic, types of files accessed, domain summary, operating system used, navigation summary and so on. We illustrate the typical Web traffic patterns of Monash University in Figures 1 (a) and (b) showing the daily and hourly volume of traffic (number of requests from different domains and the number of page requests) for the week starting 14-Jul-2002, 00:13 to 20-Jul-2002, 12:22. For more user access data logs please refer [9].

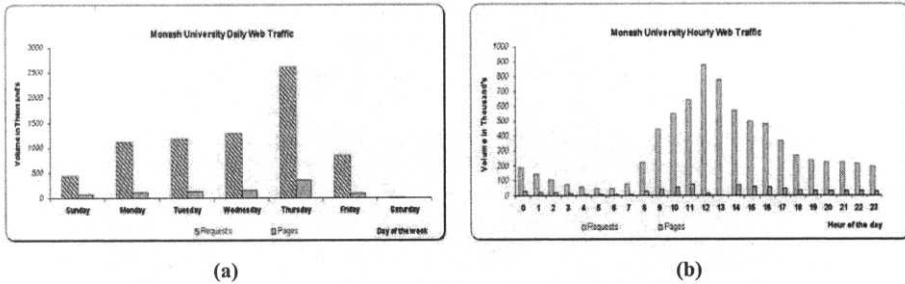


Figure 1 (a) and (b). Daily Web Traffic and Hourly Web Traffic of Monash University

In a week, over 7 million hits are received by the University's Web site [8] and since the data cover different aspects (domains, files accessed, no of daily and hourly access, page requests etc.), it is a real challenge to find some hidden information or to extract usage patterns. The mere complexity of the data volume paves way for the requirement of hybrid intelligent systems for information analysis and trend forecast. In the subsequent section, we present the structure of proposed concurrent neuro-fuzzy model for mining Web access patterns. In section 3, we present the analysis of the clustered Web data using SOM followed by modeling the TSFIS to forecast the usage pattern trends in Section 4. Finally, some conclusions and future works are given.

2. Hybrid Neuro-Fuzzy Approach for Web Traffic Mining

The hybrid framework combines SOM and a FIS operating in a concurrent environment as illustrated in Figure 2. In a concurrent model, neural network assists the fuzzy system continuously to determine the required parameters especially when certain input variables cannot be measured directly. Such combinations do not optimise the fuzzy system but only aids to improve the performance of the overall system [1]. Learning takes place only in the neural network and the fuzzy system remains unchanged during this phase. The pre-processed data (after cleaning and scaling) is fed to the SOM to identify the data clusters. The clustering phase is based on SOM; an unsupervised learning algorithm [7], which can accept input objects described by their features and place them on a two-dimensional (2D) map in such a way that similar objects are placed close together. Referring to Figure 3, data X , Y and Z may be segregated into three clusters according to the SOM algorithm. The clustered data is then used by the Web Usage Data Analyser (WUDA) for discovering different patterns and providing useful information to Web analysts.

FIS is used to forecast the Web traffic patterns on an hourly and daily basis. FIS is a popular computing framework based on the concepts of fuzzy set theory, fuzzy *if-then* rules, and fuzzy reasoning. The basic structure of the FIS consists of three conceptual components: a rule base, which contains a selection of fuzzy rules; a database, which defines the membership functions used in the fuzzy rule and a reasoning mechanism, which performs the inference procedure upon the rules and given facts to derive a reasonable output or conclusion. As shown in Figure 3, data X is associated with Cluster 3 strongly but

data Y and Z have weak associations with other clusters. Example: Data Z is associated with cluster 1 but can be also considered to have some weak association with Clusters 2 and 3. The degree of association of the data with a particular cluster is modelled as an additional fuzzy variable. We used the Takagi-Sugeno FIS to derive the rule conclusions [14]. The FIS could forecast the hourly and daily Web traffic. An important advantage of the FIS is the interpretability of the developed model in the form of simple *if-then* rules.

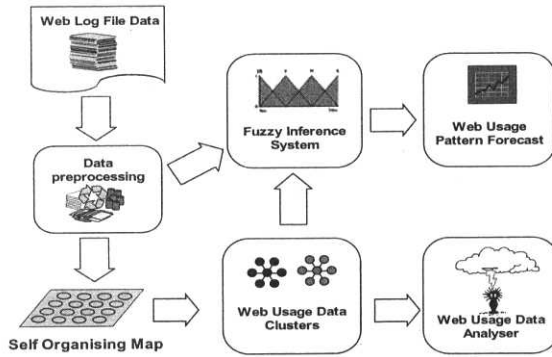


Figure 2. Architecture of the Concurrent Neuro-Fuzzy Model for Web Pattern Analysis

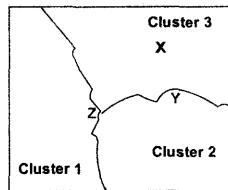


Figure 3. Data Association with Clusters

3. Data Clustering and Experimental Analysis Using SOM

The data source selected for our approach is the single-site [8] Web traffic data generated by the ‘Analog’ Web access log file analyser [3]. It is a usual practice to embed some Web trackers to analyse the user access logs. After browsing through some of the features of the best trackers available [17][16][5][3] it is easy to conclude that rather than generating statistical data and texts they really don’t help to find much meaningful information.

3.1 Data Pre-processing

In this research, we used the data from 01 January 2002 to 07 July 2002. Selecting the useful data is an important task in the data pre-processing block. After some preliminary analysis, we selected the statistical data comprising of domain byte requests, hourly page requests and daily page requests to generate the cluster models for finding Web users’ usage patterns. It is also important to remove irrelevant noisy data in order to build a precise model. Further, the datasets were scaled to 0-1. Besides the two inputs, ‘volume of requests’ and ‘volume of pages (bytes)’, we also included an additional input ‘index number’ to distinguish the time sequence of the data. The most recently accessed data were indexed higher while the least recently accessed data were placed at the bottom [2].

3.2 Data Clustering Using SOM

During working days, there are millions of user requests with different interests from different countries all around the world. The huge volume and the dynamic nature are two important features of the Web data, which invokes technology challenge to create appropriate Web mining models. SOM [7] allows different data to be grouped together

based on some similar characteristics. A 2D map of Web usage patterns with different clusters are formed after the training process. The related transaction entries are grouped into the same cluster and the relationship between different clusters is explicitly shown on the map. We used the Viscovery SOMine [15] to simulate the SOM. All the records were processed using SOM and the clustering results were obtained after the unsupervised learning process. We adopted a trial and error approach by comparing the 'normalised distortion' and 'quantization error' to decide the various parameter settings of the SOM algorithm. We finally decided the parameter setting, which could minimise both 'normalised distortion' and 'quantization' errors. The obtained 2D cluster map showing five different clusters according to the country of origin (domain) is shown in Figure 4.

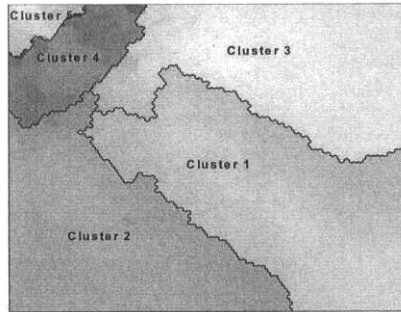


Figure 4. Clustering Results of Daily Number of Requests According to the Domains

3.3 WUDA to Discover Domain Patterns

Figure 5 depicts the details of the number of requests allocated to each cluster (total 5 clusters) according to country (domain) of origin. For clarity purpose, we have used logarithmic scale in the Y-axis.

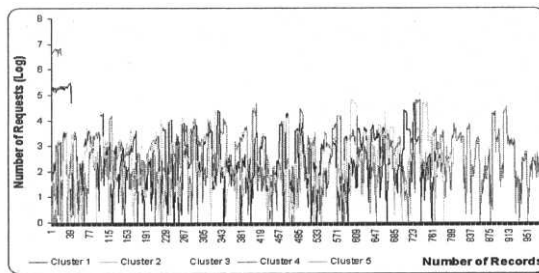


Figure 5. WUDA: Daily Number of Requests According to the Domain of Origins

As evident from Figure 5, clusters 4 and 5 separated out from the rest of the others (maximum number of requests) with a very few nodes compared to others and spread along all the time. For clusters 1, 2 and 3, the patterns are very similar with a bit of difficulty to identify each other. The domain analysis reveals that cluster 5 contains only Australian domains and cluster 4 accounts only *.com and *.net users. This is because majority of the requests originated from Australian domains (60%) followed by *.com and *.net users (8%) users. The remaining 3 clusters were shared by users from different domains depending on the volume of requests and pages (bytes). As depicted in Figure 6, more useful information is available from clusters 1, 2 and 3 with reference to the time of accessing the Web site.

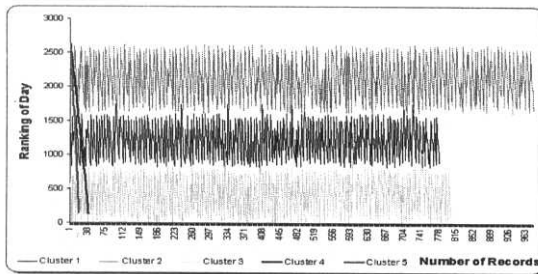


Figure 6. WUDA: Daily Number of Requests with Reference to the Time of Access

Even though clusters 1, 2 and 3 have very similar patterns for the requests, the time of access is separated very clearly as shown in Figure 6. Cluster 2 accounts for the most recent visitors and cluster 3 represents the least recent visitors. Cluster 1 accounts for the users, which were not covered by Clusters 2 and 3.

3.4 WUDA to Discover Request Patterns

The training process generated 4 clusters for the hourly number of requests. The developed clusters (indicating the hour of the day the request was made) are illustrated in Figure 7.

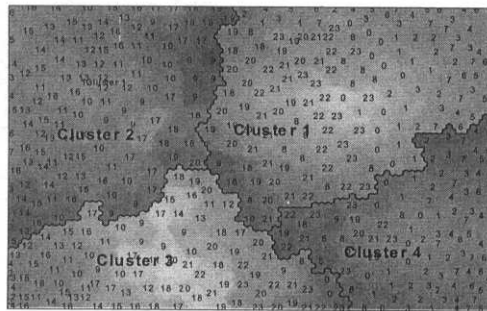
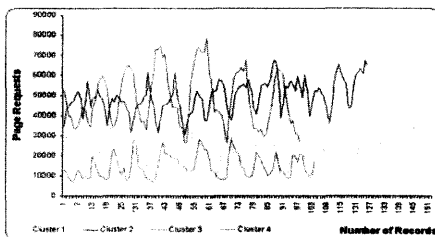
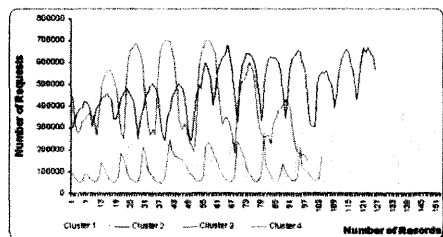


Figure 7. Hourly Requests Clusters Using SOM

From the developed clusters as depicted in Figure 7, it is very difficult to tell the difference between each cluster, as the requests (according to the hour) are scattered. From Figures 8 (a) and (b) it may be concluded that clusters 2 and 3 have much higher requests and pages (nearly double) than clusters 1 and 4.



(a) Number of hourly page requests



(b) Number of hourly requests Value

Figure 8. WUDA: Comparison of Hourly Page Traffic Volume and Page Requests

Figure 9 depicts the clustering results of the data focusing on the hourly trends of the traffic. Clusters 2 and 3 are mainly responsible for the traffic during the office hours (00:09 – 18:00) and clusters 1 and 4 account for the traffic during the University off peak hours. It is interesting to note that the access patterns for each hour could be analysed from the cluster results with reasonable classification features.

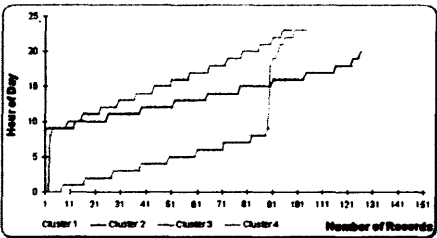


Figure 9. WUDA: Hourly Traffic Patterns

3.5 WUDA to Discover Daily Requests Clusters

Due to the dynamic nature of the WWW it is difficult to understand the daily traffic pattern using conventional Web log analysers. We attempted to cluster the data depending on the total activity for each day of the week using volume of daily requests, pages and index value as input features. The training process using SOM generated 7 clusters and the developed 2D map is shown in Figure 10.

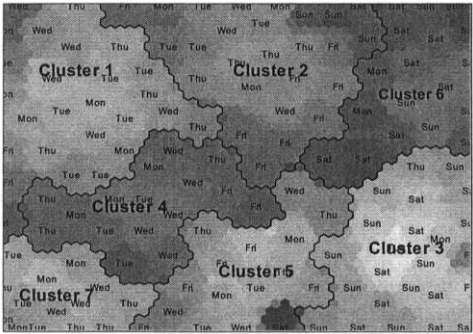


Figure 10. Developed Daily Request Clusters Showing the Days

WUDA reveals that the clusters are separated according to the time of access. Each cluster accounted the requests / access during a certain period as shown in Figure 11 (a). The ranking of the clusters are ordered as 2, 6, 1, 4, 3, 7 and 5 according to the descending order of the access time. Further analysis of the daily records in each cluster, also reveals some interesting patterns as illustrated in Figure 11 (b). Clusters 3 and 6 accounts for access records, which happened during Saturday and Sunday. Cluster 1 was separated as it only covered the first few weekdays (mostly from Monday to Thursday). While, the biggest group of clusters consist of 2, 4, 5 and 7 accounted for the transactions during Monday to Friday.

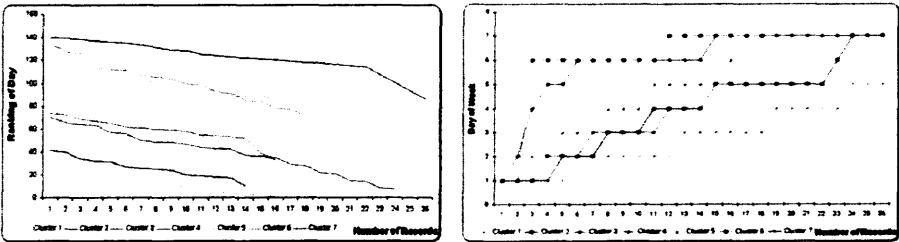


Figure 11. WUDA: Cluster Ranking Depending on "Access Time" and "Day of the Week"

4. Fuzzy Inference Systems for Forecasting

The world of information is surrounded by uncertainty and imprecision. The human reasoning process can handle inexact, uncertain and vague concepts in an appropriate manner. Usually, the human thinking, reasoning and perception process cannot be expressed precisely. These types of experiences can rarely be expressed or measured using statistical or probability theory. Fuzzy logic provides a framework to model uncertainty, human way of thinking, reasoning and the perception process. Fuzzy *if-then* rules and fuzzy reasoning are the backbone of fuzzy inference systems, which are the most important modelling tools based on fuzzy set theory. We made use of the Takagi Sugeno fuzzy inference scheme in which the conclusion of a fuzzy rule is constituted by a weighted linear combination of the crisp inputs rather than a fuzzy set [14]. A conventional FIS makes use of a model of the expert who is in a position to specify the most important properties of the process. Expert knowledge is often the main source to design FIS. According to the performance measure of the problem environment, the membership functions, rule bases and the inference mechanism are to be adapted. Evolutionary computation and neural network learning techniques are used to adapt the various fuzzy parameters [1].

In this research, we used the Adaptive Neuro-Fuzzy Inference System (ANFIS) [6] framework based on neural network learning to fine tune the rule antecedent parameters and a least mean squares estimation to adapt the rule consequent parameters of the TSFIS. A step in the learning procedure has two parts: In the first part the input patterns are propagated, and the optimal conclusion parameters are estimated by an iterative least mean square procedure, while the antecedent parameters (membership functions) are assumed to be fixed for the current cycle through the training set. In the second part the patterns are propagated again, and in this epoch, back propagation is used to modify the antecedent parameters, while the conclusion parameters remain fixed. Please refer to [6] for more details.

4.1 Design and Experimentation Results

We used the popular grid partitioning method (clustering) to generate the initial rule base. This partition strategy requires only a small number of membership functions for each input. Besides the inputs, '*volume of requests*' and '*volume of pages (bytes)*' and '*index number*', we also used the '*cluster location information*' provided by the SOM output. The data was re-indexed based on the cluster information. We attempted to develop fuzzy inference models to predict (few time steps ahead) the Web traffic volume on a hourly and daily basis. We used the data from 17 February 2002 to 30 June 2002 for training and the data from 01 July 2002 to 06 July 2002 for testing and validation purposes.

4.1.1 Daily Traffic Prediction

We used the MATLAB environment to simulate the various experiments. Given the daily traffic volume of a particular day the developed model could predict the traffic volume up to five days ahead. Three membership functions were assigned to each input variable. 81 fuzzy *if-then* rules were generated using the grid based partitioning method and the rule antecedent/consequent parameters were learned after 50 epochs. We also investigated the daily web traffic prediction performance without the '*cluster information*' input variable. Table 1 summarizes the performance of the fuzzy inference system for training and test data. Figures 12 (a), (b), (c), (d) and (e) depicts the test results for one day, two days, three days, four days and five days ahead forecast of daily Web traffic volume.

Table 1. Training and Test Performance for Daily Web Traffic Volume Forecast

Forecast Period	Root Mean Squared Error (RMSE)			
	Fuzzy Inference System (with cluster information)		Fuzzy Inference System (without cluster information)	
	Training	Test	Training	Test
1 day	0.01766	0.04021	0.06548	0.09565
2 days	0.05374	0.07082	0.10465	0.13745
3 days	0.05264	0.06100	0.12941	0.14352
4 days	0.05740	0.06980	0.11768	0.13978
5 days	0.06950	0.07988	0.13453	0.14658

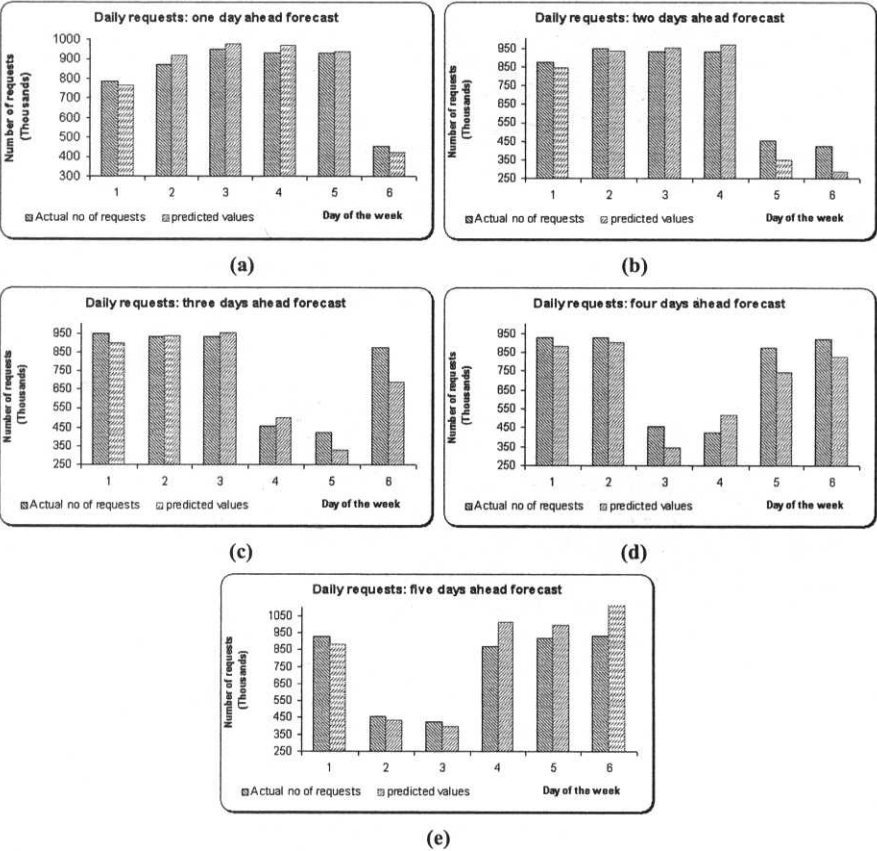


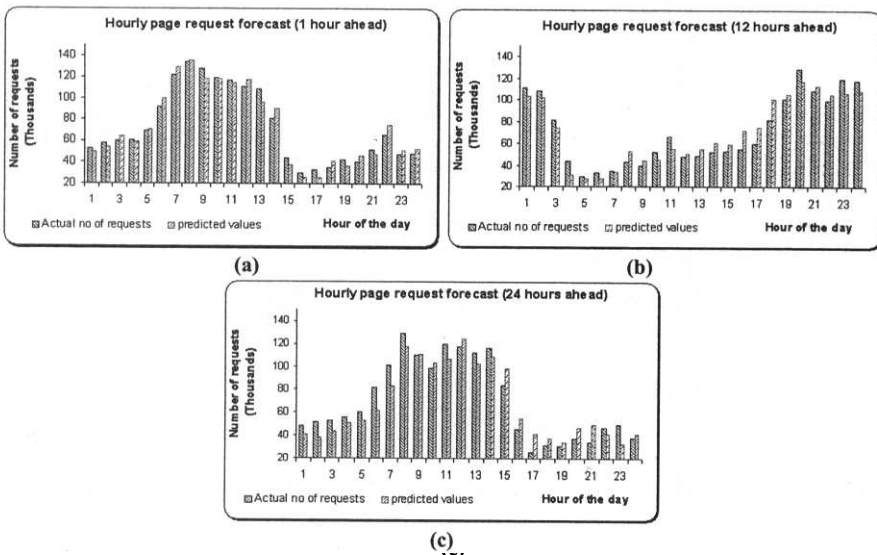
Figure 12 (a) – (e). Test Results of Daily Forecast of Web Traffic Volume

4.1.2 Hourly Page Request Forecast

Three membership functions were assigned to each input variable. 81 fuzzy *if-then* rules were generated using the grid based partitioning method and the rule antecedent/consequent parameters were learned after 40 epochs. We also investigated the volume of hourly page requests prediction performance without the ‘cluster information’ input variable. Table 2 summarizes the performance of the FIS for training and test data. Figures 13 (a), (b) and (c) illustrates the test results for 1 hour, 12 hours and 24 hours ahead forecast of the volume of hourly page requests.

Table 2. Training and Test Performance for Hourly Web Traffic Volume Forecast

Forecast Period	Root Mean Squared Error (RMSE)			
	Fuzzy Inference System (with cluster information)		Fuzzy Inference System (without cluster information)	
	Training	Test	Training	Test
1 hour	0.04334	0.04433	0.09678	0.10011
12 hours	0.06615	0.07662	0.11051	0.12212
24 hours	0.05743	0.06761	0.10891	0.11352

**Figure 13 (a) – (c).** Test results of hourly forecast of volume of page requests

5. Conclusions and Future Work

The discovery of useful knowledge, user information and access patterns allows Web based organisations to predict user access patterns and helps in future developments, maintenance planning and also to target more rigorous advertising campaigns aimed at groups of users [4]. This case study on Monash University's Web access patterns reveals the necessity to incorporate computational intelligence techniques for mining useful information. WUDA of the SOM data clusters provided several useful information related to the user access patterns. The developed FIS could predict the daily Web traffic and hourly page requests within reasonable error limits. Our experiment results also reveal the importance of the cluster information to improve the forecast accuracy of the FIS. These techniques might be useful to the website tracker software vendors to provide more useful information to the users.

We relied on the statistical/text data provided by the 'Analog' [3] software embedded at the University's Web server [9]. Analog generates the statistical data by analysing the access data logs. Due to incomplete details, we had to analyse the usage patterns for different aspects separately, preventing us to link some common information between the different aspects, trends, patterns etc. For example, the domain requests and the daily or hourly requests are all stand-alone information and are not interconnected. Therefore, a direct analysis of the Web access logs might be more helpful. We believe that if the detailed

access information could cover different interlinked features, then the usage patterns would be more comprehensive and useful. Even the access or usage patterns for particular domain within a particular time period can be analysed and predicted for marketing segment analysis and so on.

In this research, we considered only the Web traffic data during the University's peak working time. Our future research will also incorporate off-peak months (summer semesters) and so on. We also plan to incorporate more data mining techniques to improve the functional aspects of the concurrent neuro-fuzzy approach.

References

- [1] Abraham A., *Neuro-Fuzzy Systems: State-of-the-Art Modeling Techniques, Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*, Lecture Notes in Computer Science 2084, Springer-Verlag Germany, Jose Mira and Alberto Prieto (Eds.), Granada, Spain, pp. 269-276, 2001.
- [2] Aggarwal C., Wolf J.L., and Yu P.S., Caching on the World Wide Web, *IEEE Trans. On Knowledge and Data Eng.*, vol. 11, no. 1, pp. 94-107, Feb. 1999.
- [3] Analog Website Tracker, <<http://www.analog.cx/>>, (accessed on 27 July'02).
- [4] Cooley R., Srivastava J., and Mobasher B., Web Mining: Information and Pattern Discovery on the World Wide Web. In *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'97)*. 1997.
- [5] Hitbox Central Web Traffic Analysis, <<http://www.hitboxcentral.com/>> (accessed on 27 July'02).
- [6] Jang R., *Neuro-Fuzzy Modeling: Architectures, Analyses and Applications*, PhD Thesis, University of California, Berkeley, 1992.
- [7] Kohonen T., *Self-Organizing Maps*, Springer Verlag Germany, 1995.
- [8] Monash University Server Usage Statistics. <<http://www.monash.edu.au/servers/stats/main/>> (accessed on 27 July'02).
- [9] Monash University Weekly Website User Access Statistics <<http://www.monash.edu.au/servers/stats/main/httpd-log.2002-28.html>> (accessed on 27 July'02).
- [10] Ng A. and Smith K. A., Web usage mining by a self-organizing map, C. Dagli et al. (Eds.), *Smart Engineering System Design: Neural Networks, Fuzzy Logic, Evolutionary Programming, Data Mining and Complex Systems*, ASME Press, vol. 10, pp. 495-500, 2000.
- [11] Pal S.K., Talwar V., and Mitra P., Web Mining in Soft Computing Framework: Relevance, State of the Art and Future Directions. *IEEE Transactions on Neural Networks*, 2000.
- [12] Pirolli P, Pitkow J, and Rao R., Silk From a Sow's Ear: Extracting Usable Structures from the Web, In *Proceeding on Human Factors in Computing Systems (CHI-96)*, 1996.
- [13] Srivastava, J., Cooley R., Deshpande M. and Tan P.N., Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data, *SIGKDD Explorations*, 1(2): pp. 12-23, 2000.
- [14] Sugeno M., *Industrial Applications of Fuzzy Control*, Elsevier Science Pub Co., 1985.
- [15] Viscosity SOMine, <http://www.eudaptics.com/technology/somin4.html>, (accessed on 27 July'02).
- [16] Website Tracker, <<http://www.websitetracker.com/>> (accessed on 27 July'02).
- [17] WebSTAT Web Traffic Analyser, <<http://www.webstat.com/>> (accessed on 27 July'02).
- [18] Zhang Y.Q. and Lin T.Y., Computational Web Intelligence (CWI): Synergy of Computational Intelligence and Web Technology, 2002 World Congress in Computational Intelligence, IEEE Press, pp.572-575, 2002.

Panoramic View System for Extracting Key Sentences based on Viewpoints and an Application to a Search Engine

Wataru Sunayama and Masahiko Yachida

*Graduate School of Engineering Science, Osaka University,
1-3 Machikaneyama, Toyonaka, Osaka 560-8531 Japan*

Abstract.

Since there are many resources on the WWW, it has become natural for us to extract available information from them. We would like to see many documents in order to get useful information. Though summaries are useful pieces of documents, a document has various viewpoints to be summarized. Therefore, if a viewpoint of a summary is different from user's, a user cannot grasp the contents of the document correctly, and the user has to see through the documents until the end. In this paper, we present a system which makes a summary based on a user's viewpoint by user's search keywords.

1 Introduction

Nowadays, search engines are the most popular tools for seeking information on the WWW, so many support systems such as relevance feedback techniques[Rocchio 71] have been suggested. The purpose of a search is not only to seek a single piece of information, but also to gather relational information and to grasp world trends. For such purposes, we have to know the features and tendencies of existing information. Summaries of the documents are useful pieces for these purposes. Since a document has several viewpoints, if a viewpoint of a person is different from the system's, the output summary is not effective for the person to seek information. Therefore, we present a system which make a summary from a viewpoint which is given by the user.

In previous key sentence extraction methods, evaluation values are given to each sentence and sentences highly evaluated are extracted as key sentences. There are methods to evaluate sentences as follows;

1. Evaluate words in a sentence.
2. Evaluate relation among sentences.
3. Evaluate sentences by a form of the document.

As for the first method, this method corresponds to extract keywords from a document. As famous keywords extraction methods, [Luhn 58] extracts keywords by frequency of each word, and tfidf method[Salton 97] which extract keywords which appear only in the focused

document. However, it is also important the relation among the words in a sentence. Namely, the sum of the evaluation of words in a sentence won't always be correct. Although there are some methods which consider co-occurrences of words in a sentence such as [Ohsawa 02], those methods are for extracting static key sentences and do not consider user's viewpoints.

As for the second, lexical chains[Morris 91] are mainly used and a sentence related to many sentences in a document is highly evaluated[Barilay 99]. However, a sentence should contain words appeared in many sentences to be evaluated, so a sentence contains a high frequency word are tended to be extracted. Keywords should be extracted not only by its frequency but by other criteria.

As for the third, there are systems which extract initial sentences of a document, extract initial sentence of a paragraph, evaluate the place of a sentence in a document [Brandow 95] and utilize conjunctions of a document. Although these systems are useful for a particular set of documents, the methods are not general and cannot apply to Web pages because those are not written by sentences but by letters.

There is a system which supply a summary using search keywords [Tombros 98]. This system selects sentences using above first and third methods; words frequency and forms of a document. The output including both domestic key sentences and key sentences by user's viewpoints because the system doesn't use co-occurrence of search keywords and words in a document.

In this study, viewpoint keywords are made up by words which are related to search keywords by conditional probabilities of co-occurrence between search keywords and other words. Feature keywords featuring a document and also related to viewpoint keywords are extracted in words with low frequency. It is suggested that the system which extracts key sentences including both viewpoint keywords and feature keywords.

In 2, it is explained about Panoramic View System and described its algorithms. In 3, summarizations used by Panoramic View System are shown. In 4, experimental results are shown and this paper is concluded in 5.

2 Keywords Extraction by Panoramic View System

In this section, the algorithm for the Panoramic View System(PVS) is described. Though PVS is a key sentences extraction system, PVS can also evaluates viewpoint keywords and feature keywords in a document.

1. Viewpoint Keywords: Words expressing a user interest, or words strongly related to a user interest.
2. Feature Keywords: Words used only when viewpoint keywords appear and featuring a user interest.

[Algorithm for PVS]

Step 1:Input viewpoint keywords

Make an initial set of viewpoint keywords S input by the user.

Step 2:Selection of viewpoint keywords

Give a value for viewpoint keywords $view(w)$ by Eq.(1) to all the words w in a document,

and the upper ones those with values above a certain threshold which is still not included in set S are added to S . $n(w)$ is the number of sentences including a word w .

$$view(w) = \prod_{s \in S} \frac{n(w \cap s)^2}{n(w)n(s)} \quad (1)$$

This equation expresses the cross product of the conditional probability that w appears when s appears and the conditional probability that s appears when w appears. In other words, the cooccurrence of two words is normalized by the number of sentences in which each word appears. This step is repeated until no new words are added.

Step 3: Discovery of feature keywords

Find feature keywords which are connected to viewpoint keywords and which are featured in the document. The value for feature keywords $feat(w)$ is given as Eq.(2).

$$feat(w) = \prod_{s \in S} \frac{n(w \cap s)}{n(w)} \quad (2)$$

That is, find words which appear only when viewpoint keywords appear, and count how many viewpoint keywords appear in the same sentences in which those feature keywords appear.

Step 4: Evaluation of Sentences

Evaluation values are given to each sentences in a document. Each sentence T has two values given as Eq.(3) and Eq.(4).

$$view_sent(T) = \sum_{w \in T} view(w) \quad (3)$$

$$feat_sent(T) = \sum_{w \in T} feat(w) \quad (4)$$

Step 5: Key Sentences Extraction

Each sentence is evaluated by the sum of two rankings from Eq.(3) and Eq.(4) as Eq.(5). Namely, $view_rank(T)$ denotes a ranking number by the value $view_sent(T)$ and $feat_rank(T)$ is ranked by $feat_sent(T)$ ¹.

$$value(T) = \alpha view_rank(T) + \beta feat_rank(T) \quad (5)$$

α and β are parameters and are defined according to the document sets and purpose of the search². Finally, key sentences are extracted by using this value of Eq.(5). In this paper, summary is defined as sorted key sentences by the order of appearance from the original document.

¹If two sentences have the same value, both rankings are the same.

²By default, $\alpha = \beta = 1$.

Table 1: Frequency of words in this paper

Word	Frequency
sentence	44
keyword	37
are	31
system	28
word	23
viewpoint	21
search	21
document	21
extract	18
this	15

3 Experiment by summarization

We have done experiments that summarize this paper contents except for figures, tables, equations and this section.

The used viewpoint keywords are given as following;

- Experiment 1:keyword
- Experiment 2:summary

In experiment 1, a word “viewpoint” was made up by PVS for viewpoint keywords, and in experiment 2, a word “person ” was made up, too. As a story for “viewpoint keywords” is the most important topic related to “keyword”, we can say that PVS made up a proper viewpoint keyword. Similarly, “Person” means summary differs from person to person, the most relational keyword is made up as a viewpoint keyword.

Summaries for this experiments are described as following: Words typed with bold font are feature keywords, and almost all of them are appearing only once or twice. Feature keywords highly evaluated are in Table 2 and Table 3.

Table 2: Feature keywords for Experiment1 with high value

though	study	strongly	low	invisible	input	give
frequency	eq	count	connect	discovery	select	view
user's	static	make	therefore	tfidf	system's	supply
probability	focus	famous	conditional	suggest	step	since
simple	several	present	person	output	other	occasion

Table 3: Feature keywords for Experiment2 with high value

unnecessary	title	supply	sort	original	occasion	eliminate
display	comparison	piece	difine	content	apply	paper
order	purpose	document	because	system		

[Summary for Experiment 1]

Rank 5(Score:13=Viewpoint Rank 8 + Feature Rank 5)

As **famous** keywords extraction methods, [Luhn 58] extracts keywords by **frequency** of each word, and **tfidf** method[Salton 97] which extract keywords which appear only in the **focused** document.

Rank 3(Score:9=Viewpoint Rank 3 + Feature Rank 6)

In this **study**, viewpoint keywords are made up by words which relate to search keywords by **conditional probabilities** of co-occurrence between search keywords and other words.

Rank 3(Score:9=Viewpoint Rank 5 + Feature Rank 4)

[Algorithm for PVS]

Step 1:Input viewpoint keywords

Make an initial set of viewpoint keywords S **input** by the user.

Rank 1(Score:8=Viewpoint Rank 7 + Feature Rank 1)

Step 2:Selection of viewpoint keywords

Give a value for viewpoint keywords $view(w)$ by Eq.

Rank 1(Score:8=Viewpoint Rank 2 + Feature Rank 6)

Step 3:Discovery of feature keywords

Find feature keywords which are **connected** to viewpoint keywords and which are featured in the document.

[Summary for Experiment 2]

Rank 2(Score:8=Viewpoint Rank 6 + Feature Rank 2)
There is a system which supply a summary using search keywords [Tombros 98].
Rank 1(Score:7=Viewpoint Rank 6 + Feature Rank 1)
In this paper , summary is defined that sorted key sentences as the order of appearing in the original document .
Rank 2(Score:8=Viewpoint Rank 6 + Feature Rank 2)
For the comparison of the summaries, each title of Web page were not displayed .
Rank 5(Score:9=Viewpoint Rank 5 + Feature Rank 4)
That is, summaries from the all contents are needed to eliminate unnecessary information.
Rank 2(Score:8=Viewpoint Rank 3 + Feature Rank 5)
This system is effective for making summary in each occasion , because summaries of a document differ from viewpoints.

Both summaries are related to viewpoint keywords and those sentences represent this paper for the viewpoints.

4 Experiment by Search Tasks

For the outputs of a search engine, we applied summaries by Panoramic View System. The search engine used in this experiments was a simple one and results of Web pages are ranked by the number of including search keywords. The database consists of 14,500 news Web pages about public entertainment and sports.

Fig.1 is a sample output of the search engine ³. The experiment was held in Japanese and executed as follows: Each user answered questions as “When did the Sydney Olympic games open?”, by using three systems.

- 1. System A:Output initial 175 letters ⁴ of a Web page.
- 2. System B:Output 3 sentences including search keywords in the order of appearing.
- 3. Panoramic View System:Output 3 key sentences.

System A was a general search engine, and System B include search keywords but didn’t consider importance of each sentence. For the comparison of the summaries, each title of Web page were not displayed. Users were 19 people in our university who usually use search engines. Each user answered 5 questions in each and totally answered 15 different questions. 45 questions were prepared and devided into 9 sets, and difficulty of each set was averaged by previous experiments. Along with this, 11 users, selected randomly, executed this experiment twice. The number of search keywords were one from five.

The criteria of evaluation were the number of getting correct answers, the number of pushing search buttons and the number of surfing using hyper-links. Each results are shown in Table 4-Table 6 respectively. In the table, failure denotes that user inferred the answer doesn’t exist or the answer was incorrect.

There was no difference among the numbers of pushing search buttons in Table 4. However, the number of surfing in Table 5 decreases in order of System A, System B and PVS and

³Keywords in each output are viewpoint keywords, which were invisible in the experiments.
⁴175 letters are corresponding to 3 sentences.

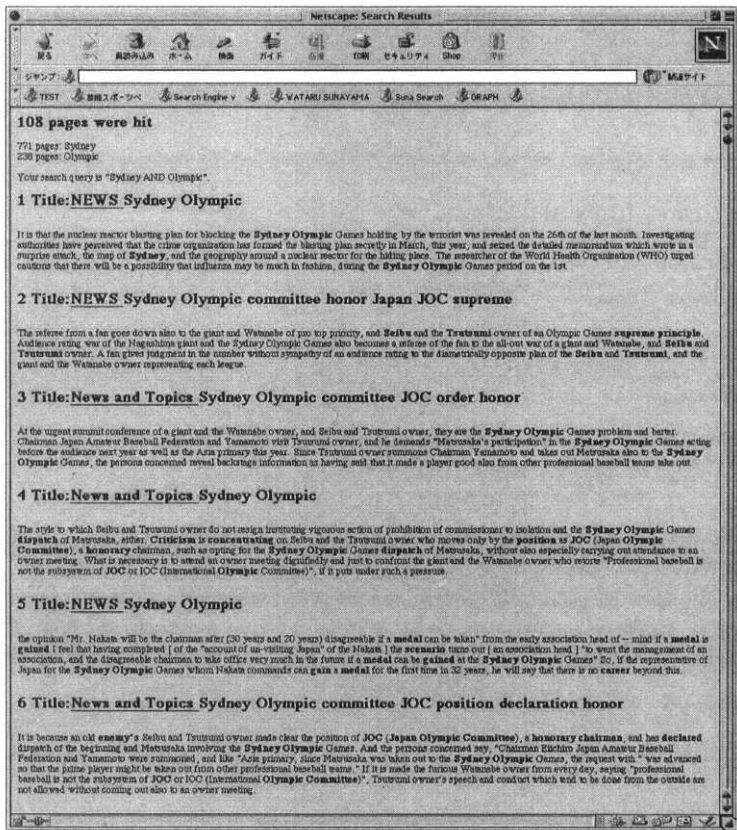


Figure 1: Search results for relational topics

Table 4: The averaged number of pushing search buttons in a question

	Success	Failure	All
System A	1.42	2.42	1.77
System B	1.31	2.24	1.63
P.V.S.	1.43	2.57	1.77

Table 5: The averaged number of surfing using by hyper-links.

	Success	Failure	All
System A	0.61	0.92	0.76
System B	0.46	0.88	0.66
P.V.S.	0.42	0.49	0.45

the number of PVS was 60-70% of System A and System B in All columns. In the case of getting correct answers, the numbers of surfing using System B and PVS are lower than that of System A. This indicates that sentences related to search keyword are necessary for discovery. In the case of failure, the number of surfing of PVS was lower than that of System A and System B, because users couldn't infer the contents by the initial part of Web pages. That is, summaries from all contents are needed to eliminate unnecessary information. Finally, as for the Table 6, the difference of getting correcting answer was caused by the difference of information obtained by the same effort.

Table 6: The number of getting correct answers

	Correct	Failure	Total
System A	97	53	150
System B	99	51	150
P.V.S.	106	44	150

5 Conclusions

In this paper, we described a Panoramic View System which extracts key sentences related to viewpoints of a document. This system is effective for making summary in each occasion, because summaries of a document differ from viewpoints. In the future, search results from multiple documents may be summarised by Panoramic View System.

References

[Barilay 99] Barilay, R. and Elhadad, M.: Using lexical chains for text summarization, *Advances in Automatic Text Summarization*, The MIT Press, London, pp.1 – 12, (1999).

[Brandow 95] R. Brandow, K. Mitze, and L.F. Rau: Automatic Condensation of Electronic Publications by Sentence Selection, *Information Processing & Management*, Vol.31, No. 5, pp. 675 – 685, (1995).

[Luhn 58] Luhn, H.P.: The automatic creation of literature abstracts, In *IBM Journal for Research and Development*, Vol.2, No.2, pp.59 – 165, (1958).

[Morris 91] Morris, J.J. and Hirst, G.: Lexical cohesion computed by thesaural relations as an indicator of the structure of text, *Computational Linguistics*, Vol.17, No.1, pp. 21 – 48, (1991).

[Ohsawa 02] Yukio Ohsawa:“KeyGraph as Risk Explorer from Earthquake Sequence”, *Journal of Contingencies and Crisis Management*,September, (2002).

[Rocchio 71] Rocchio, J.:“Relevance Feedback in Information Retrieval”, *The SMART Retrieval System - Experiments in Automatic Document Processing*, Prentice Hall Inc., pp.313 – 323, (1971).

[Salton 97] Salton, G. and Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval, *Readings in Information Retrieval*, pp.323 – 328, (1997).

[Tombros 98] Tombros, A. and Sanderson, M.: Advantages of Query Biased Summaries in Information Retrieval, In *Proc. of the 21st Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pp.2 – 10, (1998).

Frequent Flyer Points Calculator: More Than Just a Table Lookup

Grace W Rumantir

School of Multimedia Systems

Monash University – Berwick Vic 3806 Australia

grace.rumantir@infotech.monash.edu.au

Abstract

Frequent flyer program is a popular promotional tool used by most major airlines in the world. The trend of airlines forming alliances with other airlines to expand their service-base means that the potential routes that one can take to get from one city to another have increased exponentially. Airline customers however are typically provided with a table which only shows the list of direct flights or flights with a transit point that can be booked using frequent flyer points. The online frequent flyer calculators available on the internet seem to basically work based on this incomplete table. To optimise the use of available frequent flyer points and to provide satisfactory automated customer service, a better frequent flyer points calculator which accesses the complete map of the available routes and the distance of each sector of a route needs to be built. This paper shows the use of standard graph algorithms to build such a frequent flyer points calculator.

1. Motivation

Most airlines nowadays have frequent flyer program as a promotional tool which enables customers to accumulate points which can be used towards free flights (see e.g. [4]). The program usually relates the distance travelled with the number of points earned or required. As a guide, customers are provided with a table which shows the list of direct flights or flights with a transit point that can be booked using frequent flyer points. The flights are categorised into zones which determine the number of points required. A zone is defined based on a certain range of distance within which a flight covers. It does not matter whether a flight covers a distance at the lower or upper level of the range, the number of points required is the same.

On most airlines websites, a frequent flyer points calculator is usually available. It allows the user to enter the flight route he/she would like to take and the calculator will show the points required. However, all of these calculators are little more than electronic versions of the tables send out to the customers. More often than not, the calculators would print out a message to advise the user to talk directly with the airlines on routes involving multiple sectors or are partly serviced by different airlines within the alliance.

To optimise the use of available frequent flyer points, the user needs to be provided with the map of the routes that the airline services and the distance of each sector of the route. This way, the user can find out more potential routes that can be taken given the available points than those shown in the table provided. The tendency of airlines to form alliances means that there are much more potential routes that one can take; information

that currently is not readily and completely available to the public. This paper shows the use of graph algorithms to automate the painstaking process of searching for potential flight routes that a frequent flyer can take given the route maps of airlines in an alliance and their frequent flyer programs.

2. Problem Definition

The following are some of the questions that frequent flyer members of an alliance of airlines might ask in order to maximise the use of their frequent flyer points:

- 1. The minimum number of points required to fly between two cities
- 2. The potential routes that one can take given a city of origin and a number of redeemable points
- 3. The number of points earned and required for a given multi-sectored route

A useful frequent flyer points calculator would produce a report showing the detail of each sector of the route, such as:

- The city of origin, destination city and the names of the airports
- The distance covered
- The number of points required/earned on the airline servicing the sector

Finally, the report gives a summary of the total points required/earned on individual airlines for a multi-sectored route serviced by a number of airlines.

2.1. Case Study

To build a Frequent Flyer Points Calculator that can satisfactorily answer the above questions, we form a fictitious alliance of three airlines: Billabong Air, Ukingdom Air and Yankee Air. The alliance enables their frequent travellers to use their frequent flyer points to fly free on the networks serviced by the three airlines. It is assumed that the networks of the three airlines do not overlap.

2.1.1. Charts for Redeeming and Earning Points

A frequent flyer program has rules on how points are earned from paid trips and used for award trips. In our case study, the three airlines have their own frequent flyer award programs which structure is typically shown in Table 1.

Table 1. A typical example of a frequent flyer award chart. In our case study, the three airlines have different award programs with the same structure.

Zone	Maximum km Distance	Points Required		
		Economy	Business	First
1	0 – 1,2400	9,000	11,250	13,000
2	1,2401-1,700	17,000	21,250	25,500
3	1,701 – 2,900	25,000	31,250	37,500
4	2,901 – 7,300	30,000	37,500	45,000
	. .etc .			
10	Over 28,980 km	150,000	225,000	300,000

To calculate the number of points earned for a flight, a formula like the one shown in Table 2 is also commonly available.

Table 2. A typical example of a frequent flyer points earning chart. If a customer is a member of frequent flyer programs of more than one airlines in the alliance, the customer has to chose which program the points earned for a paid trip should be added to.

Points per km	Fare
1.50	First Class
1.25	Business Class
1.00	Full Economy
0.70	Excursion/Discount Economy

2.1.2. Distance Between Two Airports

Every airport in the world has a unique code and world coordinate positions. So, whilst the routes in the network serviced by an airlines is subject to change depending on business climate, the flight sector between two airports serviced by an airline can be uniquely identified.

The world coordination position of an airport is expressed in latitude (South to North) and longitude (East to West) in degrees, minutes and seconds. Given the position of two airports, L_1 , N_1 and L_2 , N_2 , and the equatorial radius of the earth, $R = 6378$ km, the distance between the two airports in radians can be calculated using the approximated formula called the Great Circle Distance given in Equation 1. To use the formula, the unit of the positions of the two airports should first be converted to positive radian.

$$Distance = arccos[\sin L_1 * \sin L_2 + \cos L_1 * \cos L_2 * \cos(N_2 - N_1)] * \pi/180 * R \quad (1)$$

Using Equation 1, the distance of the flight sectors in the network of an airline can be calculated and used to calculate the frequent flyer points earned from or required for a trip.

3. Problem Solving

The problem of finding routes in a flight network can be seen as an application of weighted undirected graphs covered in a Data Structures and Algorithms subject in a Computer Science course. In graph terminology, cities/airports are *nodes* (or *vertices*), a direct flight between two airports is an *arc* (or *edge*) and the distance of a direct flight is the arc's *weight*. Since the connection between two airports is always two ways, the graph representing the network of an airline is said to be *undirected*.

The network of an airline can be stored in the computer system for the purpose of building the frequent flyer calculator by setting up two files:

1. Airport Master file which stores the detail of each airport in the network.
Each record comprises:
 - a. A unique airport code
 - b. The name of the airport
 - c. The city in which the airport is located
 - d. The location of the airport in latitude and longitude coordinate
2. Connection file which stores the direct flights in the network of an airline.
Each record represents a direct link between an airport of origin and a destination airport which comprises:
 - a. The code of the airport of origin
 - b. The code of the destination airport
 - c. The distance between the two airports. This is calculated using Equation 1 based on the locations of the two airports given in the Airport Master file.

Because there is no overlap in the networks of the airlines in the alliance, the connection files can be loaded up in memory in two ways:

- In one *adjacency matrix* with the cities of origin as the rows and the destination cities as the columns and the distance and the airline that service each sector as the values in the cells. Effectively this matrix is a symmetric matrix with zeros in the diagonal cells.
- In a set of *adjacency lists* with the city of origin given at the head of each list followed by the destination cities directly connected with the city of origin.

Given the network and the charts in Table 1 and 2, the questions posed in Section 2 above can be answered by first finding the distance of a route using the appropriate graph algorithms and then map the distance into the points charts. Each question will be discussed in the following subsections.

3.1. Minimum number of points required to fly between two cities

Redeeming the minimum number of points for a free flight between two cities in a network means finding the shortest route between the two cities. Once the shortest route has been found then the points required for the route can be found by looking up the points chart in Table 1. In this example the points required for each airline is summed up separately, but in real life, airline alliances often have a separate points chart for trips which require the services of multiple airlines in the alliance.

Dijkstra's shortest path algorithm can be used to find the shortest path between two points in a graph. With the networks represented in an adjacency matrix `distanceMatrix`, the shortest path algorithm [2] can be shown as follows:

FindShortestRoute (float distanceMatrix[][MAXCITIES], integer cityOfOrigin, integer cityOfDest, float shortestDistance, int shortestRoute[])

```
{
    int KNOWN = 1; UNKNOWN = 0; // if min distance from origin is known
    float INFINITY = MAX_FLOAT
    integer minDistanceFromOriginToCityKnown[MAXCITIES];
    float minDistance[MAXCITIES];

    int latestCityWithKnownMinDistance;
    int i, k, dc;
    int smallDistance, newDistance;

    /* initialisation */
    for (i = 0; i < MAXCITIES; ++i) {
        minDistanceFromOriginToCityKnown[MAXCITIES] = UNKNOWN;
        minDistance[MAXCITIES] = INFINITY;
    } // end for

    // at present, only the min distance from cityOfOrigin to cityOfOrigin is known
    minDistanceFromOriginToCityKnown[cityOfOrigin] = KNOWN;
    minDistance[cityOfOrigin] = 0;
    latestCityWithKnownMinDistance = cityOfOrigin;

    while (latestCityWithKnownMinDistance != cityOfDestination) {

        smallDistance = INFINITY;
        dc = minDistance[latestCityWithKnownMinDistance];

        // check all cities for successor of latestCityWithKnownMinDistance

        for (i = 0; i < MAXCITIES; ++i)

            if (minDistanceFromOriginToCityKnown[i] == UNKNOWN) {

                newDistance =
                    dc + distanceMatrix[latestCityWithKnownMinDistance][i];

                /* Distance from cityOfOrigin to i
                 * through latestCityWithKnownMinDistance is shorter than
                 * minDistance[i] */
                if (newDistance < minDistance[i]) {
                    minDistance[i] = newDistance;
                    shortestRoute[i] = latestCityWithKnownMinDistance;
                } // end if

                // determine the shortest distance */
                if (minDistance[i] < smallDistance) {
                    smallDistance = minDistance[i];
                    k = i;
                } // end if
            } // end if

        latestCityWithKnownMinDistance = k'
        minDistanceFromOriginToCityKnown[latestCityWithKnownMinDistance] = KNOWN;
    } // end while

    shortestDistance = minDistance[cityOfDestination];

} // end FindShortestRoute
```

Figure 1 shows screenshots of the function to find the minimum number of points required to fly between two cities using the implementation of the FindShortestRoute algorithm and the points charts which structure is shown in Table 1.

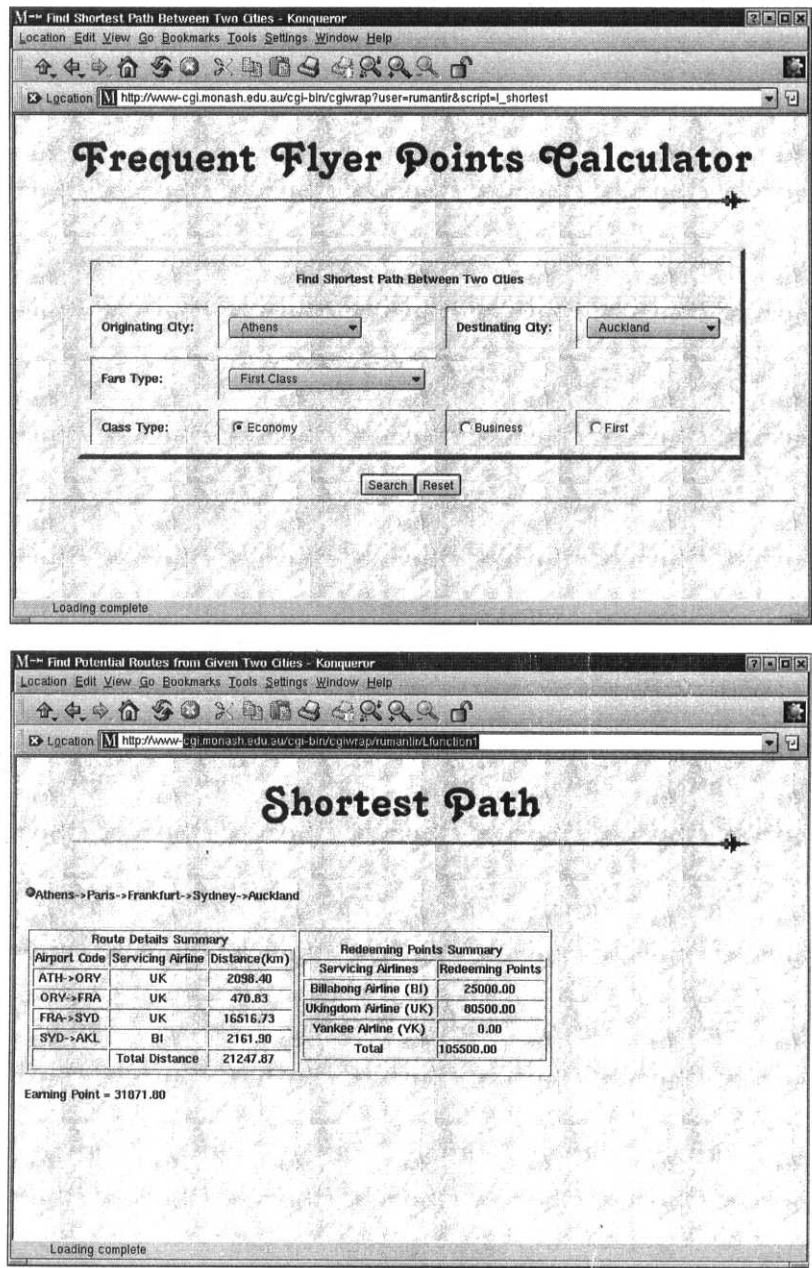


Figure 1. Finding the minimum number of points required to fly between two cities

3.2. Potential routes given a city of origin and redeemable points

Finding all of the potential routes from a city of origin means traversing and recording all of the unbroken links in the joint network. Each traversal can stop once the cost to travel the route equals or exceeds the available redeemable points.

Depth-first or breadth-first algorithms can be used to find all of the potential routes using either an adjacency matrix or a set of adjacency lists. The following algorithm illustrates the depth-first algorithm [1] using an adjacency list of the joint network.

```
findAllRoutesFromOrigin (int city)
{
    visited[city] = TRUE;

    printf("%5d", city);

    for (successorCity = graph[successorCity]; successorCity = successorCity -> link)
        if (!visited[successorCity -> vertex])
            findAllRoutesFromOrigin (successorCity -> vertex);
}
```

Figure 2 shows screenshots of the function to find all of the potential routes given a city of origin and a number of redeemable points using the implementation of the `findAllRoutesFromOrigin` algorithm and the points charts which structure is shown in Table 1.

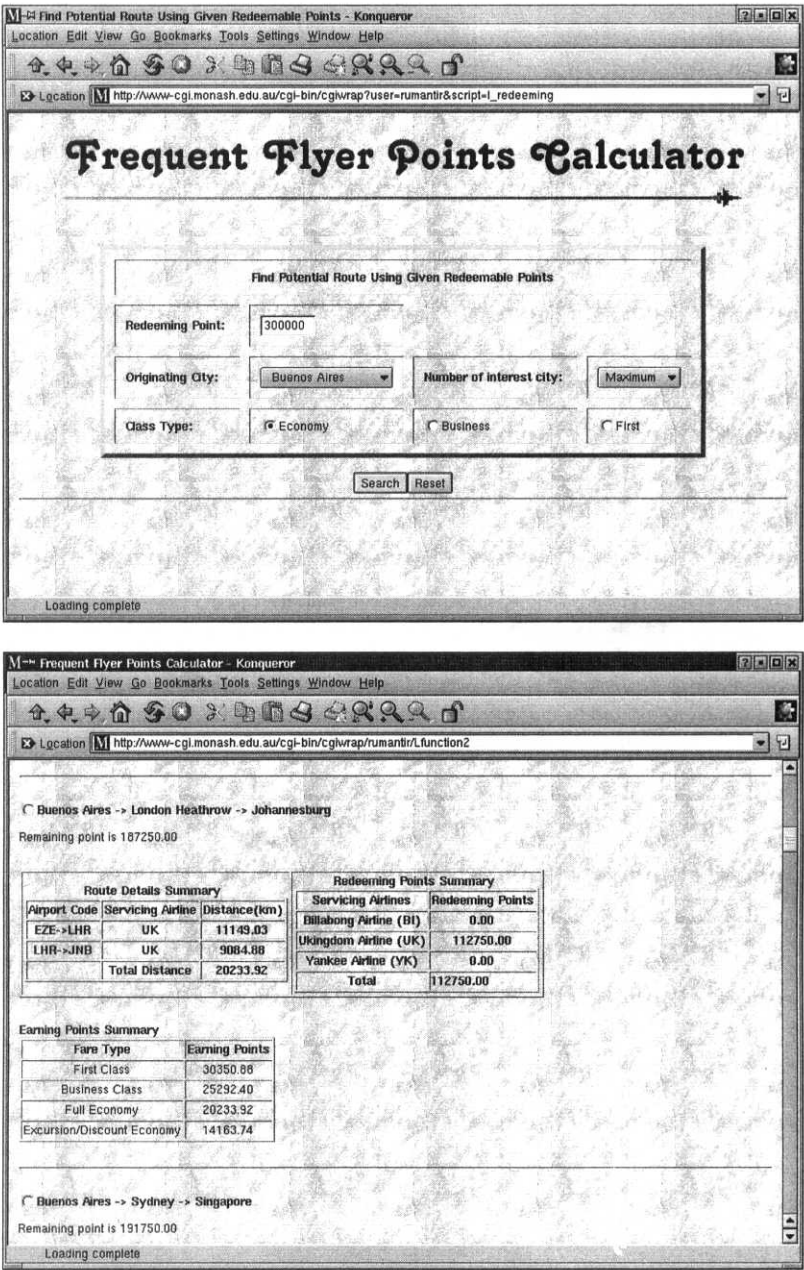


Figure 2. Finding all potential routes given a city of origin and redeemable points

3.3. Points earned and required for a given multi-sectored route

This function can be made computationally very efficient by designing the graphical user interface such that the user is presented with just the direct routes from each city of origin in the form of a combo-box or a pull-down list. This way, the calculations of the points earned or required are just a matter of summing up the distance between two adjacent cities shown in respective cell of the adjacency matrix.

Figure 3 shows the screenshots of the input and result screens of the function.

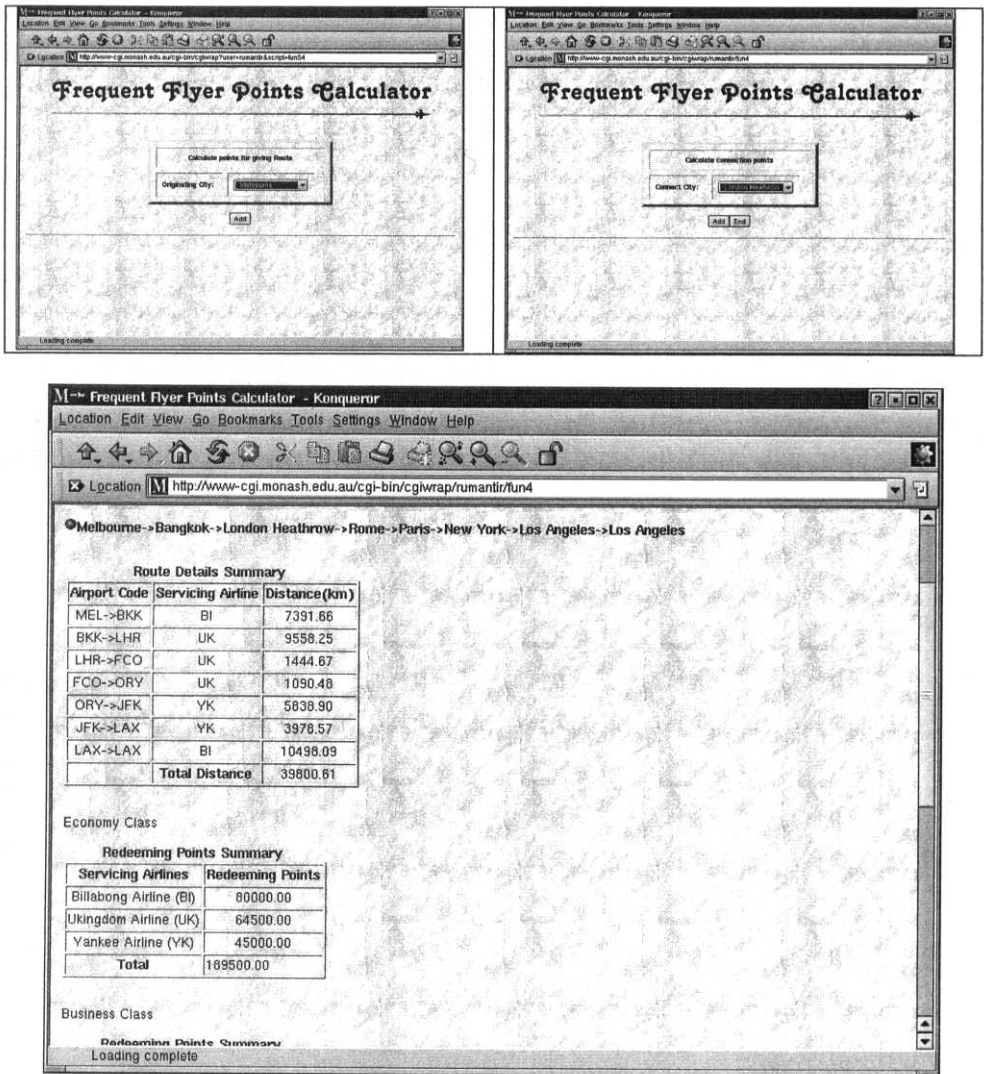


Figure 3. Points earned and required for a given multi-sectored route

Conclusions

By using standard graph algorithms and effective user-friendly GUIs as illustrated in this paper, airline customers are given more freedom and options to make informed decisions on how to maximise the use of their frequent flyer points and how to earn the maximum points for a trip they plan to make within the boundary of the rules set by the airlines.

This will create a win-win situation both for the customers as well as the airlines because the customers will require less assistance from the airline staff to make their best decisions hence reducing the cost for customer service for the airlines. By being able to automatically show the customer complete stopover options along the way in their trip, the customer is likely to take advantage of the stopovers which is good for the tourism industry as a whole.

Acknowledgments

This case study has been used as a Software Engineering project for third year students within the School of Computer Science and Software Engineering Monash University 1999. The author would like to thank Susanti, Goh Yin Chiat, Viyada Tarapornsin, Yoke Ching Yee, Prinya Vechbanyongratana and Mei Ki Amy Yung for implementing the approach proposed in this paper. The screenshots used in this paper are produced using their program.

References

1. E. Horowitz, S. Sahni, S. Andreson-Freed, *Fundamental of Data Structure in C*, Computer Science Press, 1993
2. Y. Langsam, M. J. Augenstein, A. M. Tenenbaum, *Data Structures Using C and C++*, Second Edition, Prentice-Hall, 1996
3. G. W. Rumantir, *CSE3302 Software Engineering Project Specifications*, School of Computer Science and Software Engineering, Monash University, 1999.
4. Qantas Frequent Flyers Web Page, www.qantas.com

A Large Benchmark Dataset for Web Document Clustering

Mark P. Sinka, David W. Corne

Department of Computer Science, University of Reading, Reading, RG6 6AY, UK
m.p.sinka@reading.ac.uk, d.w.corne@reading.ac.uk

Abstract: Targeting useful and relevant information on the WWW is a topical and highly complicated research area. A thriving research effort that feeds into this area is *document clustering*, which overlaps closely with areas usually known as *text classification* and *text categorisation*. A foundational aspect of such research (which has been proven over and over again in other research disciplines) is the use of standard datasets, against which different techniques can be properly benchmarked and assessed in comparison to each other. We note herein that, so far in this broad area of research, as many datasets have been used as research papers written, thus making it difficult to reason about the relative performance of different categorisation/clustering techniques used in different papers. In this paper we propose a standard dataset with a variety of properties suitable for a wide range of clustering and related experiments. We describe how the dataset was generated, and provide a pointer to it, and encourage its access and use. We also illustrate the use of part of the dataset by establishing benchmark results for simple *k*-means clustering, comparing the relative performance of *k*-means on a pair of 'close' categories and a pair of 'distant' categories. We naturally find that performance is better on the pair of 'distant' categories, however the experiments reveal that although stop-word removal is confirmed as helpful, word-stemming is, (perhaps counter to intuition), not necessarily always recommended on 'distant' categories.

1. Introduction

Most available search engines primarily function by providing a list of documents contained on the World Wide Web that contain matches to given keywords and/or phrases. However, keyword matching is known to be only suggestive of a document's relevance, and improvements to keyword matching need to be found in order for the World Wide Web to reach its full potential. To this end, considerable research is now being invested into more sophisticated ways to analyse and assess the content of web documents. For example, if the World Wide Web can be clustered into different subsets and labelled accordingly, search engine users can then restrict their keyword search to these specific subsets.

It seems fairly clear that we cannot expect manual classification to be achievable. The precise size of the World Wide Web is unknown and is growing all the time, however what is known is that Google [1] claims to index over 2 billion web documents, and according to [2], over 1.5 million web documents are added to the World Wide Web every day. Human back-classification of at least 2 billion web documents would be virtually impossible, and even attempting to categorize all new web documents would require unacceptable human effort. One possible solution is to

force web document authors to categorize each newly created page. This has three problems; firstly web authors cannot be relied upon to categorize their pages correctly, secondly authors are often prone to misclassify their documents in order to increase potential web traffic, and thirdly it does not address the problem of the (at least) 2 billion web pages that have already been created. Therefore researchers in this field are turning to autonomous, or semi-autonomous methods for web document categorization ([2], [3], [4], [5], [6], [7], [8], [9], [10]).

There are many other potential applications and benefits that will accrue from being able to reliably and *automatically* cluster and categorize corpora of documents. Much of this *document clustering* work is based on using either supervised or unsupervised learning techniques in order to label particular web documents as belonging to a specific category, or grouping together similar documents into clusters. This research area closely overlaps with (and in recent times indistinguishable from) research efforts known as *text classification* and *text categorization*. Van Rijsbergen [11] has carried out seminal work in this area, while excellent modern surveys have been done in [12] and [13], while [14] also provides a helpful tutorial.

Various techniques have been proposed that aim to develop accurate methods for autonomous categorization. However, the research literature in this area soon reveals that almost every single such proposed technique has been tested for its categorization accuracy using different datasets; any objective, scientifically sound comparison between two categorization techniques is therefore very difficult. This issue has also highlighted in [14]. Whilst it may be appealing to infer that a technique *X* that is shown to have a considerably superior accuracy than technique *Y* on dataset *A* would be similarly superior on dataset *B*, this is not really so clear cut. For example, stop-word removal [11] and stemming [15] are almost universally done to reduce the sizes of feature vectors for documents, in the belief that little information of use is lost (see, e.g., [6]); however, [16] shows counter-evidence to the general truth of this (as do we, later on). The essential point is that more widespread use of well-designed common datasets in this area will help elucidate all of the factors involved. For example, suppose dataset *A* is composed of two clusters which are relatively easy to separate, while dataset *B* may contain two very similar clusters, which are much harder to separate, and whatever strategy made technique *X* perform so well on dataset *A* may simply not be useful on dataset *B*. In the more general case, it is of course very hard to make general statements about two techniques that produce results that are *close*, in terms of accuracy, if they have not been tested on the same dataset.

The main aim of this paper is to propose a dataset for general use in web document clustering and similar experiments; the design, content, generation and location of this dataset are described in section 2. We note that this dataset cannot be specifically ‘proven’ to be optimally useful for web document clustering investigations, but it was designed with a view towards making it as flexible and useful as possible for such research and related work. In addition to the generation of a benchmark dataset, it is also important to establish benchmark values against which the accuracy of future techniques can be measured. Our particular research interest is the unsupervised clustering of web documents. Unsupervised clustering is attractive in this problem domain because, unlike supervised learning, it does not require a training set or domain expert in order to learn. This is important because firstly it is extremely difficult for any human or groups of humans to generate even a partial taxonomy for the World Wide Web, secondly any such taxonomy would be highly debatable and subject to question, and thirdly the fact that new categories emerge frequently and others diminish in importance would require such a taxonomy to be continually

updated. We are particularly interested in the ability of unsupervised methods to separate ‘close’ datasets, and in this paper we describe baseline experiments using straightforward k -means clustering, aimed at beginning to understand any interactions between the ‘distance’ of the categories being clustered, and aspects of the design of simple document feature vectors. Our k -means implementation, associated issues, and the experimental set-up are described in section 3, while the results of these experiments are listed in section 4. We briefly summarise and conclude in section 5.

2. A Benchmark Dataset

The datasets used by previous authors have varied a lot in terms of size and content. A dataset containing as few as 1,000 web documents was used [7], whereas in [8] 20,000 documents were used. The size of a dataset is important especially when considering unsupervised learning techniques such as k -means clustering, due to factors such as the increased sensitivity to initial cluster centres with a smaller dataset. Also there are many problems that are associated with web document feature extraction, such as the document representation, that are not as apparent in smaller datasets, which of course are less like the real world applications that we are attempting to address.

Human categorization of a dataset of sufficient size would take too long; hence researchers have employed previous human classification where it has been available. Humanly categorized sets of related web sites are called Web Directories, with the most common being Open Directory Project [17], Yahoo! Categories [18], and LookSmart [19]. Previous authors’ datasets, despite taking advantage of these directories, have not extracted the data in the same way. The dataset generation method used in [7] involved picking 200 documents from 5 of the 14 top-level categories in Yahoo! [3]. This method of random web document selection not only meant that a high proportion of very small web documents were brought back, but also, as acknowledged in the paper, some documents belonged to multiple categories. This confounds human categorization, let alone automatic categorization.

Another important issue to consider is the number of categories contained within the dataset. A dataset comprising just 2 distinct categories would be much easier to partition into two clusters than would a dataset comprising 10 different categories, even if the latter dataset’s categories fell into two clear themes. Our approach, as we see below, is to use a relatively large number of categories, but with sufficient of each category to allow a wide variety of sensible experiments, each using clearly defined subsets of the dataset.

Finally, as another argument towards the use of a standard dataset, we note that datasets used so far in document clustering research vary greatly in content. The same comparative experiments on two different datasets with different content, irrespective of size or the number of categories contained within, may easily produce conflicting results. The need for a common dataset that will allow for the accuracy of different techniques to be benchmarked is clear.

2.1 Dataset Design

The first distinct design decision was to ensure that each page in our dataset had a least a small amount of content. The task of autonomous categorization of medium to large

web documents is very complex and challenging, so it seems unwise to make this task even harder by attempting to classify documents containing very little or no information. Therefore our proposed dataset is biased towards pages that actually have some content, so a dataset that has as few 'very small' pages as possible should make developing and training of a classification/clustering system easier. It should be noted, therefore, that this dataset is not a realistic 'snapshot' of the World Wide Web.

The number of pages required for a useful, multi-purpose, dataset means that human classification of each web page in our dataset would be far too expensive in terms of time and effort. Therefore we made use of the Open Directory Project [17] and Yahoo! Categories [18] to provide web pages that have already been humanly categorized. Much thought then went into the selection of our dataset categories. The goal was to create a dataset consisting of some sets of categories quite distinct from each other, as well as other categories that were quite similar to each other. This allows for a range of clustering issues to be researched using the same dataset. The inclusion of similar categories also ensures that more complex partitioning tasks can be performed therefore fully testing the categorization accuracy of a particular method.

A suitable balance of categories was achieved by first selecting two broad but very distinct themes, namely "Banking & Finance" and "Programming Languages", and picking out three sub-categories from each of these two themes. The resulting six categories are: "Commercial Banks", "Building Societies", "Insurance Agencies", "Java", "C / C++" and "Visual Basic". It should be immediately clear that the task of partitioning these six categories into two groups, (Finance and Programming Languages) should be much easier than partitioning all the programming languages into three groups, which itself is easier than partitioning all of the six combined categories into six different groups. This therefore achieves our goal of generating a dataset allowing for various partitioning tasks with varying difficulty.

Four more categories were then selected, to widen the potential use and varieties of experiments that could be performed with this dataset. Since the themes "Banking & Finance" and "Programming Languages" are quite distinct, we decided it would be wise to include two more broad themes, one close to an existing theme, and one that was totally distinct from all categories so far. We therefore chose "Science" because it is *somewhat* close to "Programming Languages", and "Sport" because it is quite distinct from all the other categories in the database. The extra four categories which make up the main ten in our database were therefore: "Astronomy", "Biology", "Soccer" and "Motor Sport".

Finally, we felt it useful to add an extra category that was in some way a 'parent' of two existing ones. This provides a good test for hierarchical clustering methods, and for generally reasoning about the results of clustering in terms of hierarchical relationships among the data. The extra dataset is "Sport", and contains web documents from all the sites that were classified as sport (in [17] and [18]), but with the sites that occurred in either the "Soccer" or "Motor Sport" datasets removed. This allows us to think of quite complex partitioning tasks, such as the task of partitioning the union of sets "Sport", "Soccer" and "Astronomy" into two groups. A full list of the 11 main dataset categories is shown in Table 1.

In order to determine the number of documents required to make our dataset useful we looked at not only the size of dataset used by previous researchers, but the number of documents per category also. Knowing that our dataset consisted of 11 categories, the decision was whether to download and archive 100, 200, 500, 1,000 or 2,000 documents per category. We felt that 1,000 documents per category would

suffice, therefore giving an overall dataset size of 11,000 documents. The next subsection details how we went about extracting the pages themselves.

Table 1 – Dataset categories and their associated themes

Dataset Id	Dataset Category	Associated Theme
A	Commercial Banks	Banking & Finance
B	Building Societies	Banking & Finance
C	Insurance Agencies	Banking & Finance
D	Java	Programming Languages
E	C / C++	Programming Languages
F	Visual Basic	Programming Languages
G	Astronomy	Science
H	Biology	Science
I	Soccer	Sport
J	Motor Sport	Sport
K	Sport	Sport

2.2 Page Selection

For each of the first 10 main categories in table 1, we extracted the set of all websites contained within the associated category listing in the Open Directory Project [17], and combined them with the set of all websites contained within the associated category listing on Yahoo! [18]. The only information stored about each web site was the entry page, which allowed our web spider to crawl the rest of the site. Each site in the database was then crawled, noting down the size and URL of each page. The size of a page was determined by first removing any scripts, style-sheets, or comments beforehand. Each site's URLs were then ordered, in descending size of the referenced page, and stored in the database. The next step was to remove any sites that contained fewer than 10 pages in total. This was done because of the way in which the URLs are archived.

Table 2 – Attributes saved within each web document

Id number
Archiving date and time
Page size
Corresponding Category (Content label)
HTML Source

The archiving involved looping through each web site in the database and 'popping' off the largest web page's URL from that site. The contents of the URL were then retrieved and examined for the presence of a frameset. If the page was, or contained a frameset then the rest of the frames pointed to were added to the current page before archiving. This ensured the actual content (as seen by a human) was archived. Each complete page was then archived with the content indicated in table 2.

We freely encourage the use of the dataset and a complete version can be downloaded from: <http://www.pedal.reading.ac.uk/banksearchdataset/>

3. Baseline Experiments with *K*-Means Clustering

Some experiments were done which are now described; these experiments had two aims. First, to establish some baseline results against which future techniques applied to the same dataset can be evaluated. Second, these experiments form the beginning of our study into appropriate techniques for the unsupervised clustering of documents that are contained in ‘close’ categories. This is a particularly interesting problem which relates to, among other things, technologies which can automatically discern fine-grained differences within document sets and (for example) add extra value to the output of search engines.

We use only the most basic, although popular and effective in many areas, unsupervised clustering technique, *k*-means, and also use a basic approach to developing feature vectors to represent the documents (simple word-frequency vectors). These choices served both of the aims as follows. First, more sophisticated techniques can only be properly evaluated if results are available for simpler methods on the same data – indeed, simple *k*-means may well offer the best speed/performance trade-off on certain datasets. Second, although a number of sophisticated and interesting techniques are currently under study in unsupervised document clustering ([20], [21], [22], [10]), this field is far less researched so far than *supervised* document categorisation. This heightens the importance, in our view, of thorough study of rudimentary techniques in the field, especially given the lack of dataset standardisation so far, and the fact that little or nothing seems to have been done concerning the unsupervised separation of close categories.

Towards understanding *k*-means performance on ‘close’ category discrimination, we did two sets of experiments, both with *k* fixed appropriately at 2. One set used *k*-means to separate two semantically very similar categories of documents: sets *B* and *C* (see Table 1). The other set of experiments used *k*-means to separate two quite distinct categories: sets *A* and *I* (see Table 1). In all cases, the full 1,000 documents in the dataset for each of the two categories were used. Hence, each experiment attempted to cluster 2,000 documents. In each of the two main sets of experiments, we did 10 trial runs for each of 16 different feature vector configurations as follows. The feature vector representing a document (as described next in more detail) is a simple vector of scaled word frequencies. However, the vector was built either with or without stemming [15], and either with or without removal of stop-words [11]. Also, the vector was built only using the top *h*% of words (in the 2,000 documents being clustered) in terms of frequency of occurrence. We tested four values of *h*: 0.5, 1, 1.5, and 2. Results are presented in section 4, but before that we give more detail of the experimental set-up and our *k*-means implementation.

3.1 Feature Extraction

Stressing simplicity first, our feature vectors were built only from text that would be seen on the screen, i.e. normal document text, image captions and link text, and no

extra weight was given according to emphasis (bold typeface, italic typeface, different colours, etc ...). For each document, the extraction process was as follows:

- The set of all words that appeared at least once in the document was extracted.
- If stop-word removal was switched on, we removed from the set of extracted words (Step 1) any word that was listed in our stop-word list, (we used Van Rijsbergen's list [11]).
- If word stemming was switched on, we combined all the words with a similar stem, (i.e. count all occurrences of a word as a single occurrence of its stem).
- We then recorded and stored the frequency of each word in that document.

Once all the documents in the chosen set of categories were thus processed, the next step was to create a master word list that containing every word in the combined dataset, associated with its overall frequency. Then we cut down the master list to contain only the top $h\%$ of most frequently occurring words, where h was varied between experiments. Finally, a feature vector \mathbf{v}_i was created for each document i , such that the j^{th} element in \mathbf{v}_i was w_{ji}/s_i , where w_{ji} is the number of occurrences in document i of the j^{th} most frequent word in the combined dataset, and s_i is the total number of words in document i .

3.2 Clustering

Our implementation of k -means clustering was standard, although certain issues tend to vary between implementations and we clarify those here. K -means begins by generating random vectors to act as initial cluster centres. Each required cluster centre was created by coping the contents of a randomly chosen document feature vector from the vector space. Another important issue is the treatment of 'dead' cluster centres (containing no documents, since all vectors are closer to some other cluster centre). We chose to do nothing when a cluster centre died, in case a vector became re-assigned to it in the future. Again, this stressed simplicity, although other options are generally more favourable, especially when k is low.

4. Results

The results are summarised in Table 3. For each configuration, three measurements are given – these are the mean, median, and best accuracies over ten trials for that configuration. We measure the accuracy of a single clustering run as the percentage of documents that were classified correctly. That is, if 2-means clustering happens to separate the A & I dataset into two clusters of 1,000 documents, with all the A in one cluster and all the I in the other, that represents 100% accuracy. More generally, a result would place X documents from a category into one cluster, and the remaining Y of those documents into the other cluster.

Table 3– Experiment Results: mean, median and best of ten runs for two sets of sixteen experiments each using different configurations of frequency vector size, stop-word removal, and stemming. A and I are the dissimilar datasets and B and C are similar datasets (see Table 1). The highest mean, median and best for each experiment is underlined, and the best accuracy for each set (A & I, B & C) is in bold type.

Datasets	Stemming / Stop-word removal	Accuracy	Size of word-frequency vector			
			0.5%	1%	1.5%	2%
A & I	No/No	Mean	57.12	56.31	<u>59.64</u>	53.2
		Median	56	50.3	<u>64.9</u>	50.77
		Best	64.95	<u>66.4</u>	66.3	66.3
A & I	No/Yes	Mean	62.38	66.65	<u>71.12</u>	58.8
		Median	51.07	50.875	<u>71.7</u>	50.025
		Best	90.65	92.55	93.0	<u>93.05</u>
A & I	Yes/No	Mean	59.8	58.27	<u>60.51</u>	58.75
		Median	59.2	50.8	<u>60.0</u>	50.8
		Best	69.4	71.1	71.15	<u>71.6</u>
A & I	Yes/Yes	Mean	53.93	65.31	54.25	<u>67.51</u>
		Median	50.15	50.75	50.05	<u>55.6</u>
		Best	87.1	87.8	<u>91.95</u>	91.35
B & C	No/No	Mean	53.66	<u>53.79</u>	50.68	50.6
		Median	<u>51.9</u>	51.8	50.65	50.05
		Best	<u>59.9</u>	58.2	51.7	51.7
B & C	No/Yes	Mean	<u>53.67</u>	53.52	51.44	52.36
		Median	<u>54.3</u>	<u>54.3</u>	51.6	53.03
		Best	<u>54.7</u>	54.3	52.75	54.4
B & C	Yes/No	Mean	<u>53.7</u>	51.01	50.69	51.55
		Median	<u>52.1</u>	51.0	50.15	51.95
		Best	<u>62.15</u>	52.1	51.95	51.95
B & C	Yes/Yes	Mean	70.9	<u>75.44</u>	64.19	69.22
		Median	74.02	<u>89.0</u>	56	60.9
		Best	89.1	90.05	89.2	<u>90.35</u>

We then took the cluster containing the larger number of documents to be the correct cluster, and calculated accuracy on that basis. Many experiments yielded poor results in which all or nearly all documents ended up in a single cluster, which nevertheless scores 50% on this accuracy measure.

Throughout the experimental trials, there were many poor results, owing to the sensitivity of (straightforward) k-means to the positioning of the initial random cluster centres. This is apparent in the low mean and median values; such sensitivity was particularly problematic in those experiments which show low medians (near 50%), which indicate that more than half of the ten trials places all documents within a single cluster. However, sufficient trials were done to show interesting effects as follows.

probabilistically.

By effectively sharing/disseminating information among individuals, swarm strategies [14] incorporate mechanisms to solve optimization problems. This approach incorporates species' seeds which form the nucleus of a collection of random individuals. Each individual swarm member communicates with its nearest neighbors (NN) in order to improve their performance and obtain the information required in searching for the optimum among/in smaller groups (also known as afterswarms in honeybee terminology).

Selection pressure on subpopulations is reduced when the "migration interval" for an application is random. This approach facilitates implementation of the methodology of the SBGA [19] algorithm. The continuous creation of subpopulations leads to various combinations of individuals with similar genetic makeups. This results in shifts in the gene space, which, in turn, facilitate the exploration of the disparate regions of the search space.

An ant colony optimization (ACO) approach [13, 10] for the resource constrained project scheduling problem (RCPSP) was implemented to simulate the pheromone trails associated with each ant. The use of the pheromone trail required the implementation of heuristics that determine the rate of pheromone decay (evaporation).

7 Conclusion

The creation of an artificial ecosystem within the Internet provides an adaptive search model which has the ability to evolve and keep pace with the Internet's evolution and further development. The exponential growth of related Web documents presents IR problems for existing and newly developed engines. The IR problems are magnified as the existing engines set new goals for incorporating documents that do not comprise the group of pages considered the core of Internet related documents. The communication patterns within this simulated system benefit from the adaptive communication and hierarchical habits of honeybees.

8 Acknowledgments

The author wishes to thank D. Stott Parker and Gary B. Fogel for their direction and suggestions. The author wishes to express his gratitude to the reviewers whose detailed and useful comments helped tremendously to improve the quality of this paper. This work was supported by Honeybee Technologies and Tapicu, Inc.

References

- [1] H.A. Abbass. MBO: Marriage in Honey Bees Optimization A Haplometrosis Polygynous Swarming Approach. In *Proceedings of CEC 2001*, pages 207–214. IEEE, Piscataway, NJ, 2001.
- [2] R. Bagrodia. Process Synchronization: Design and Performance Evaluation of Distributed Algorithms. *IEEE Transactions on Software Engineering*, 15(9):1053–1064, September 1989.
- [3] P.J. Bentley, T.G.W. Gordon, J. Kim, and S. Kumar. New Trends in Evolutionary Computation. In *Proceedings of CEC 2001*, pages 162–169. IEEE, Piscataway, NJ, 2001.
- [4] M. Conrad and H.H. Pattee. Evolution Experiments with an Artificial Ecosystems. *Journal of Theoretical Biology*, 28:393–409, 1970.
- [5] D.B. Fogel. An Introduction to Simulated Optimization. *IEEE Transactions on Neural Networks*, 5(1):3–14, 1994.

References

- [1] Google Search Engine, <<http://www.google.com>>
- [2] Pierre, J.M. (2000) *Practical Issues for Automated Categorization of Web Sites*, September 2000. <<http://citeseer.nj.nec.com/pierre00practical.html>>
- [3] Boley, D., Gini, M., Gross, R., Han, S., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., Moor, J. (1999a) Document Categorization and Query Generation on the World Wide Web Using WebACE, *AI Review*, 13(5-6): 365-391,
- [4] Boley, D., Gini, M., Gross, R., Han, S., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., Moor, J. (1999b) Partitioning-Based Clustering for Web Document Categorization, *Decision Support Systems*, 27(3): 329-341, <<http://citeseer.nj.nec.com/9105.html>>
- [5] Goldszmidt, M., Sahami, M. (1998) *A Probabilistic Approach to Full-Text Document Clustering*, Technical Report ITAD-433-MS-98-044, SRI International, <<http://citeseer.nj.nec.com/goldszmidt98probabilistic.html>>
- [6] Moore, J., E. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, and B. Mobasher. (1997) Web page categorization and feature selection using association rule and principal component clustering. In *7th Workshop on Information Technologies and Systems*, Dec 1997. <<http://citeseer.nj.nec.com/moore97web.html>>
- [7] Tsukada, M., Washio, T. and Motoda, H. (2001) Automatic Web-Page Classification by Using Machine Learning Methods, in N. Zhong, Y. Yao, J. Liu, S. Oshuga (eds.) *Web Intelligence: Research and Development, Proceedings of the 1st Asia Pacific Web Conference on Web Intelligence*, LNAI 2198, Springer-Verlag, Berlin, pp. 303-313
- [8] Wong, W., Fu, A.W. (2000) Incremental Document Clustering for Web Page Classification, *IEEE 2000 Int. Conf. on Info. Society in 21st century: emerging technologies and new challenges*, Nov 5-8, 2000, Japan. <<http://citeseer.nj.nec.com/article/wong01incremental.html>>
- [9] Zamir, O., Etzioni, O. (1998) Web document clustering: A feasibility demonstration. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 46-54, Melbourne, Australia.
- [10] Zhao, Y., Karypis, G. (2002) *Criterion Functions for Document Clustering: Experiments and Analysis*, <<http://citeseer.nj.nec.com/zhao02criterion.html>>
- [11] Van Rijsbergen, C.J. (1975) *Information Retrieval*, Butterworths.
- [12] Aas, K., Eikvil, A. (1999) *Text Categorisation: A survey*, Technical report, Norwegian Computing Center, June, <<http://citeseer.nj.nec.com/aas99text.html>>
- [13] Sebastiani, F. (1999a) *Machine learning in automated text categorisation: A survey*. Technical Report IEL-B4-31-1999, Istituto di Elaborazione dell'Informazione, C.N.R., Pisa, IT, 1999. <<http://citeseer.nj.nec.com/article/sebastiani99machine.html>>
- [14] Sebastiani, F. (1999b) A Tutorial on Automated Text Categorisation. In Analia Amandi and Ricardo Zunino, editors, *Proceedings of ASAI-99, 1st Argentinean Symposium on Artificial Intelligence*, pages 7-35, Buenos Aires, AR, 1999. <<http://citeseer.nj.nec.com/sebastiani99tutorial.html>>
- [15] Porter, M.F. (1980), An algorithm for suffix stripping, *Program*, 14(3): 130-137.
- [16] Riloff, E. (1997) Little words can make a big difference for text classification. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 130-136. <<http://citeseer.nj.nec.com/riloff95little.html>>
- [17] Open Directory Project, <<http://www.dmoz.org>>
- [18] Yahoo! Directory, <<http://www.yahoo.com>>
- [19] LookSmart, <<http://www.looksmart.com>>
- [20] Tishby, N., Pereira, F.C., Bialek, W. (1999) The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, 368-377, 1999. <<http://citeseer.nj.nec.com/tishby99information.html>>
- [21] Slonim, N. and Tishby, N. (2000) Document clustering using word clusters via the information bottleneck method. In *Proceedings of the 23rd International Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 208-215.
- [22] Slonim, N., Tishby, N. (2001) The power of word clusters for text classification. In *23rd European Colloquium on Information Retrieval Research*. <<http://citeseer.nj.nec.com/slalom01power.html>>

Simulating an Information Ecosystem within the WWW

Reginald L. Walker
Tapicu, Inc., P.O. Box 88492
Los Angeles, California 90009
rwalker@tapicu.com

Abstract. The design focus of the Tocarime Apicu integrated search engine builds upon new approaches and techniques associated with evolutionary computation to improve the precision and recall mechanisms of existing information retrieval systems within popular search engines. The interactions of the four major components of engines are facilitated through the use of a hierarchical communication topology which partitions the nodes of a distributed computing system into subclusters. The hierarchical communication topology is based on an information ecosystem modeled upon and incorporating the social structure of honeybees—this providing mechanisms for the efficient sharing of information.

1 Introduction

The information sharing/communication model associated with Tocarime Apicu¹ research effort [18, 17, 16] incorporates aspects of the unique behavior exhibited by inhabitants of a honeybee colony [9, 11] in relation to their external environment, including their relation to other honeybee colonies. These aspects are employed to adequately search and index portions of the Web for valuable information by viewing the WWW as an information ecosystem. Ultimately, resulting in a comprehensive event manager (EM) model [2], the honeybee model [14] removes communication limitations inherent in current methodologies by providing the basis for information sharing mechanisms. This model can be extended by treating each subcluster—based on the nodes associated with each manager, M_i , in the EM model—as a set of queens and drones.

Various techniques are employed to continuously disperse foragers within the honeybee information sharing model to serve the needs of an existing colony in their search for the location of ever-changing food sources (time-dependent information) which are prone to change drastically over a relatively short period of time in a manner similar to requesting information from a remote site within the Internet as shown in Figure 1. Web page retrieval is accomplished within the Tocarime Apicu engine by the HTML Resource Discovery (HRD) system [16] and Web page parsing by the Information Sharing Indexing (ISI) system [18, 17]. The foragers mark food sources as well as the path to the food source in order to formulate customized routes [16, 13, 10]. This methodology has the ability to discover new ISPs as well

¹The word *Tocarime*, meaning “spirit”[7], comes from an ancient Amazon Indian language. *Apicu* comes from the Latin *apis cultura*, meaning “honeybee culture” or “the study of honeybees.” The phrase *Tocarime Apicu* is used as “in the spirit of bee culture.”

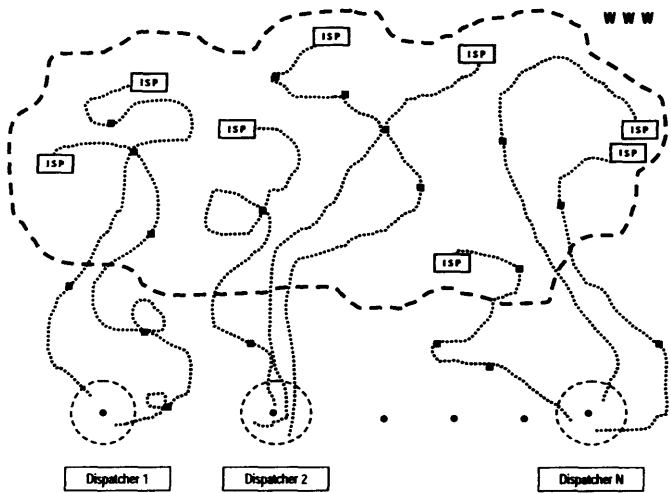


Figure 1: The WWW as an information ecosystem as viewed by the Tocarime Apicu HRD system Web dispatchers.

as new sub-host providing services to new and existing Web clients which result in faster discovery of new and updated Web pages.

2 The Problem

Most of the current engines [12] provide its users with varied results—this depending on the type of producer—this in turn leading to the coupling of link analysis, human-based categories, and automated spidering technology. The result is overlapping pages/information. Several of these engines are producers as well as consumers of pre-compiled sets of Web pages. The unique qualities of crawler-based systems coupled with the results of human-editors provides an assortment of information sources, this further aggravating the task of indexing for the consumer engines which attempt to eliminate page duplication in the query results. Inevitably, this process should lead to diversity which may be apparent in the query results associated with various search strings.

The use of link analysis—which is based on the Web pages having been clicked/accessed by various past users of particular engines—allows various servers to handle different categories reflective of regional user interest. Such an approach is the underlying focus of Google [8] where the shortcomings of link analysis [12] are based on Web page connectivity which has been predicted as belonging to one of these categories:

- 1. Core pages—30 percent of the Web pages are considered interconnected and easy to find by their Internet connection links.

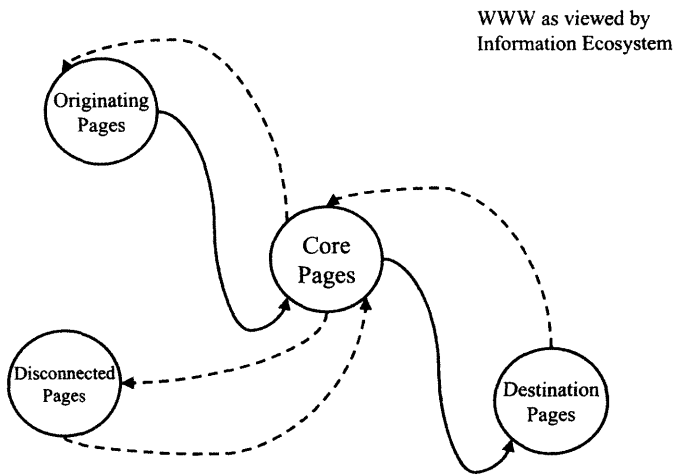


Figure 2: Web page connectivity as viewed by the Tocarime Apicu information ecosystem. Solid lines show Web page connectivity as viewed by Google.

2. Originating pages—24 percent of the Web pages cannot be accessed from the core pages but are linked to them.
3. Destination pages—24 percent of the Web pages can be accessed from the core pages but are not linked to them.
4. Disconnected pages—22 percent of the Web pages are disconnected from the core pages.

3 The Simulate Honeybee Ecosystem

3.1 The Information Ecosystem

The creation of an information ecosystem, which emulates selected aspects of distributed honeybee colonies, incorporates three distinct hierarchical levels of biological organization [4, 5, 19] using the methodologies of evolutionary computation (EC) [17] and EM: the population, the organismic, and the genetic levels. The EM model provides the means by which the various interactions within and between levels of the ecosystem can take place.

The restrictions placed on the ecological system were: 1) the process of evolutionary search must agree with biological facts, even if non-biological search techniques are more effective, and 2) the information ecosystem must be as simple or primitive as possible. These restrictions correspond to the evolutionary processes incorporated in biological species, enabling them to solve problems typified by chaos, chance, temporality, and nonlinear inter-

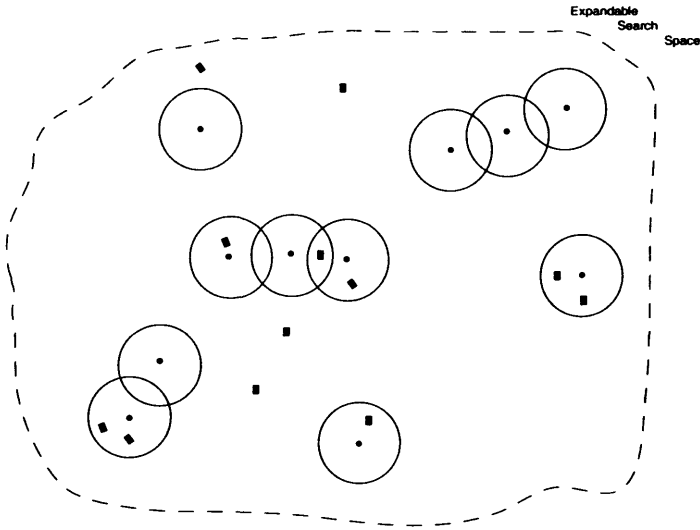


Figure 3: A snapshot of the expandable search space for overlapping NNCs as viewed by the ISI system.

activity [5, 3]. Figure 2 presents Web page connectivity as viewed by the Tocarime Apicu information ecosystem.

3.2 The Hierarchical Levels of Honeybees

The social hierarchy of honeybees [11, 9] results from environmental influences—ecological and physiological, genetic potentiality, and social conditions (regulations) of the colony. Social insects are characterized by 1) cooperation among adults in brood care and nest construction, 2) overlapping of at least two generations, and 3) reproductive division of labor—queen specializes in egg laying and workers in brood care. The evolutionary search strategy of this system initiates unique characteristics of the honeybee model—its spatial organization along with species specificity, this in turn influencing the pattern of gene flow, and therefore the search strategy.

The methodology of EC provides mechanisms to facilitate the requirements needed to model the organic evolution of purposive behavior [6] which incorporates various aspects of honeybees. Organic evolution requires: 1) a system of stored information, 2) a system of transferring information from one generation to the next, 3) a system of mutation of stored information, and 4) a system of translations of the genetic information into a form against which value judgments are made. Purpose behavior has two aspects which are: 1) the ability to specify an intrinsic purpose and 2) elaborating a pattern of behavior that is directed towards achieving a specified purpose. The evolution of purposive behavior is achieved by requiring 1) stochastic fluctuations to break the impasse of indeterminacy and 2) the actions associated with the selection operator. The outcome of this process is a deterministic sequence of events resulting in a stochastic process which sets a basically indeterminate system into one of many possible pathways.

Table 1: Cumulative access log summaries for Web probe dispatchers.

Item	Web probe dispatchers
Start date	15 Oct 2001
Stop date	28 Jan 2002
Duration in days	105 days
Duration in weeks	15 weeks
Simultaneous probe transmissions	128
Total requests	48193332.0
Avg requests per day	458984.1
HTML servers located	75367

4 Overview of the Tocarime Apicu Information Sharing (IS) System

4.1 Web Page Indexing Components

Restricted and free breeding were implemented by the formulation of an expandable search space (Figure 3) which forms adaptive subclusters. The search space shown in the figure depicts a snapshot of the location of drifter nodes with respect to seed nodes. The seed nodes are represented as small circles and the drifters as small rectangles. Each specie seed represents the basis of a subgroup if it exists. A subgroup is formed when one or more drifters are within a given radius of a seed forming nearest neighbors. Each seed uses the same radius. An extension of this approach better reflects the biological model, allowing the seed radius to increase/decrease based on the decay of its pheromones [13, 10]. Supersedure emulation occurs when two or more seeds are nearest neighbors forming overlapping nearest neighbor clusters (NNCs). When a single drifter is a nearest neighbor of two disjoint seeds, this node may be selected to share information twice based on the existence of one or more competing drifters.

4.2 Web Foraging Components

The Web probes, scouts, and foragers are responsible for retrieving external data. The raw external data files are reduced to raw HTML files in the first pass that contain no padding or HTML header information [15], such as date of last modification or date to expire. The resulting raw HTML file is equivalent to the information presented to the user by her/his chosen Web browser. The raw external file contains information that can be used by the advanced mechanisms, which, themselves are components of some popular browsers.

The second pass on the raw HTML file occurs during the tag parsing process; this is a component of the ISI system. The data to be used in these mechanisms is filtered during the first pass in order to assure that the appropriate wrappers exist. Wrappers used by the HRD foragers filter out documents that may not be in English or follow incorrect HTML format. Currently, the search engine is unable to perform any language translations. The HRD dispatchers communicate with the ISI dispatchers via their respective group managers. Likewise, the ISI dispatchers communicate with the browser reporting interface (BRI) dispatchers through their respective manager interfaces.

Table 2: Stochastic measures associated with Web probe dispatchers.

Week	f_{loc}	f_{succ}	$P(M,i)$	$R(z)$	$I(M,i,z)$
1	0.04548	0.01734	0.00004	115127.0	230253.9
2	0.03687	0.01312	0.00003	153503.4	460510.1
3	0.03962	0.01610	0.00004	115127.0	460507.8
4	0.04210	0.01705	0.00004	115127.0	460507.8
5	0.04194	0.01595	0.00004	115127.0	690761.7
6	0.04078	0.01361	0.00003	153503.4	1074523.6
7	0.04025	0.01309	0.00003	153503.4	1228027.0
8	0.04184	0.01595	0.00004	115127.0	1036142.6
9	0.05023	0.01819	0.00003	153503.4	1535033.7
10	0.03905	0.01463	0.00005	92101.1	1013112.1
11	0.04024	0.01502	0.00004	115127.0	1381523.4
12	0.04035	0.01683	0.00005	92101.1	1197314.4
13	0.04244	0.01702	0.00005	92101.1	1289415.5
14	0.04215	0.01687	0.00004	115127.0	1726904.3
15	0.03919	0.01599	0.00004	115127.0	1842031.2
Mean	—	—	—	120755.5	1041771.3
SD	—	—	—	21612.3	475118.8

5 Experimental Results

5.1 HRD Experimental Environment

The HRD system searched the Internet for those ISPs hosting Web services [17] for a total of fifteen weeks which included three U.S. holidays—Thanksgiving, Christmas, and New Years (see Table 1). The start date was 15 October 2001 and the terminating date was 28 January 2002 with one-week data collection periods that span from Monday to Monday.

The goal of this study was to test the run-time environment associated with dispatchers and determine the limitations in executing the HRD network probing software for extended periods of time. The HRD system was tested using HP Pavilions with seven 733MHz (20 Gigabytes of memory) and one 766 MHz (30 Gigabytes of memory) Intel Celeron processors, 128 MB SDRAM, and Intel Pro/100+ Server Adapter Ethernet cards, connected via two D-Link DSH-16 10/100 dual speed hubs with switches through a 144 Kbps router. The dispatcher tests were run using Red Hat Linux release 7.0 (Guinness).

5.2 Web Probe Dispatchers

Usage of stochastic measurements [16] provides insight into the efficiency of the system and thus reflects performance degradations. The localized fitness, f_{loc} , of each version of the probe dispatcher per week is presented in Table 2. The number of independent runs required to achieve an efficiency of 99%, $R(z)$, showed moderate variations. The standard deviation for $R(z)$ and $I(M, i, z)$ was 21612.3 and 475118.8, respectively. The values associated with f_{loc} are dependent on the number of probes per week, thus reflecting the collisions within the LAN and TTL associated with the released probes. The success fitness, f_{succ} , which is also based on the weekly sum of all four dispatchers, determines the effectiveness of locating ISPs

Table 3: Web scout dispatcher results for customized routing.

ISP response results	Web scout dispatcher				Totals
	Node 0	Node 1	Node 2	Node 3	
Number of DNS name resolutions (dispatcher result code Y)	9262	6774	6821	5870	28727
Number of QoS requests (based on RS statistic plots)	9502	9180	9587	9326	37595
Periodic tasks (recurring QoS requests)	240	2406	2766	3456	8868
Number of customized routes (successful retrievals)	1422	1127	1003	1112	4664

that host HTML services.

The corresponding number of projected dispatchers, $I(M, i, z)$, is not computed as a value independent of the future collection period. The value projects the number of dispatchers required to correct shortcomings of the current dispatchers. The number of required weeks, $R(z)$, needed to achieve the targeted probability was 92.1 thousand weeks for collection periods 11, 13, and 14. A projected value of 115.1 thousand weeks is required for collection periods 2, 4, 5, 6, 9, 12, 15, and 16. The remaining collection periods required 153.5 thousand weeks. The number of dispatchers needed for collection period 2 was projected as 0.2 million. Collection periods 3, 4, and 5 will require 0.5 million dispatchers. Collection periods 7, 8, 9, 11, 12, 13, and 14 will require between 1.0 and 1.5 million dispatchers. Collection periods 10, 15, and 16 will require between 1.5 and 1.9 million dispatchers. These results show some consistency between the distinct collection periods. This suggests that increasing the LAN throughput to the Internet should reduce all projected collection periods and dispatcher requirements.

5.3 Web Scout Dispatchers

The RS statistic results in the Table 3 differ from the references used as the basis of this approach [16] since the larger RS statistics value implies a smaller degree of self-similarity. RS statistics measurements are periodically used to determine the degree of congestion along the path to a chosen ISP, as well as within the chosen ISP. The retrieval of raw data files removes the ISP from the lists of periodic tasks which are then retested. The feasibility tests are based on the release of scouts to each ISP in order to perform the self-similarity computations. A congestion threshold was used to determine feasibility results, indicating 4544 customized routes. All of the customized routes contained at least one scout that did not return within its allocated TTL. These time-outs reflect possible congestion in the LAN, LAN+WAN, or WAN.

The scout dispatcher results are shown in Table 3. The total number of DNS name resolutions 28727. The increase is due to the extended collection period and dispatcher improvements [16].

Table 4: ISP responses to Web foragers inquiries based on customized routing.

Week	ISP response results	Web forager dispatcher				Totals
		Node 0	Node 1	Node 2	Node 3	
Week 1-15	Number of customized routes (retrieval of raw data files)	9262	6774	6821	5870	28727
Week 16	Raw HTML pages	690	536	548	557	2331
	Access forbidden pages					
	—Firewall pages	21	27	44	31	123
	—Web mail pages	110	144	129	155	538
	—403 forbidden pages	4	3	5	4	16
	—404 not found	0	0	2	2	4
Week 17	Useful raw HTML pages	555	362	368	365	1650
	Raw HTML pages	732	591	455	555	2333
	Access forbidden pages					
	—Firewall pages	75	10	28	17	130
	—Web mail pages	214	144	216	173	747
	—403 forbidden pages	3	1	0	1	5
Totals	—404 not found	1	1	0	1	3
	Useful raw HTML pages	439	435	211	363	1448
	Raw HTML pages	1422	1127	1003	1112	4664
	Access forbidden pages					
	—Firewall pages	96	37	72	48	253
	—Web mail pages	324	288	345	328	1285
	—403 forbidden pages	7	4	5	5	21
	—404 not found	1	1	2	3	7
	Useful raw HTML pages	994	797	579	728	3098

5.4 Web Forager Dispatchers

Use of the customized routes produced raw data files which were not always useful. The raw HTML pages were the results of the file being validated by pre-parsing the raw contents, which are a component of the ISI system. The pre-processing may result in NULL files which were discarded, thus indicating that either the HTML format was incorrect, or that the designer used non-English tags. Other non-useful pages indicated a firewall or Web mail login (access forbidden) HTML page but were retained in these studies.

Table 4 presents the corresponding ratios: 15.4%, 16.6%, 14.7%, and 18.9% for the forager dispatchers, respectively. The cumulative dispatcher ratio for this version was 16.2%. The results reflect the use of filters that eliminated a host of raw HTML pages through implementation of the first pass of the two-pass parser [18] required by the Tocarime Apicu ISI system.

6 Related Work

A search algorithm termed “optimization with marriage in honeybees” (MBO) [1] emulated the mating behavior of bees. The simulated behavior in this approach relied on the in-flight aspect of the mating-flight of queens. The variable-speed flight of the queen was viewed as a set of transitions in a state space in which the queen mates with a different drone at each state

probabilistically.

By effectively sharing/disseminating information among individuals, swarm strategies [14] incorporate mechanisms to solve optimization problems. This approach incorporates species' seeds which form the nucleus of a collection of random individuals. Each individual swarm member communicates with its nearest neighbors (NN) in order to improve their performance and obtain the information required in searching for the optimum among/in smaller groups (also known as afterswarms in honeybee terminology).

Selection pressure on subpopulations is reduced when the "migration interval" for an application is random. This approach facilitates implementation of the methodology of the SBGA [19] algorithm. The continuous creation of subpopulations leads to various combinations of individuals with similar genetic makeups. This results in shifts in the gene space, which, in turn, facilitate the exploration of the disparate regions of the search space.

An ant colony optimization (ACO) approach [13, 10] for the resource constrained project scheduling problem (RCPS) was implemented to simulate the pheromone trails associated with each ant. The use of the pheromone trail required the implementation of heuristics that determine the rate of pheromone decay (evaporation).

7 Conclusion

The creation of an artificial ecosystem within the Internet provides an adaptive search model which has the ability to evolve and keep pace with the Internet's evolution and further development. The exponential growth of related Web documents presents IR problems for existing and newly developed engines. The IR problems are magnified as the existing engines set new goals for incorporating documents that do not comprise the group of pages considered the core of Internet related documents. The communication patterns within this simulated system benefit from the adaptive communication and hierarchical habits of honeybees.

8 Acknowledgments

The author wishes to thank D. Stott Parker and Gary B. Fogel for their direction and suggestions. The author wishes to express his gratitude to the reviewers whose detailed and useful comments helped tremendously to improve the quality of this paper. This work was supported by Honeybee Technologies and Tapicu, Inc.

References

- [1] H.A. Abbass. MBO: Marriage in Honey Bees Optimization A Haplometrosis Polygynous Swarming Approach. In *Proceedings of CEC 2001*, pages 207–214. IEEE, Piscataway, NJ, 2001.
- [2] R. Bagrodia. Process Synchronization: Design and Performance Evaluation of Distributed Algorithms. *IEEE Transactions on Software Engineering*, 15(9):1053–1064, September 1989.
- [3] P.J. Bentley, T.G.W. Gordon, J. Kim, and S. Kumar. New Trends in Evolutionary Computation. In *Proceedings of CEC 2001*, pages 162–169. IEEE, Piscataway, NJ, 2001.
- [4] M. Conrad and H.H. Pattee. Evolution Experiments with an Artificial Ecosystems. *Journal of Theoretical Biology*, 28:393–409, 1970.
- [5] D.B. Fogel. An Introduction to Simulated Optimization. *IEEE Transactions on Neural Networks*, 5(1):3–14, 1994.

- [6] A.S. Fraser. The Evolution of Purposive Behavior. In H. von Foerster, J.D. White, L.J. Peterson, and J.K. Russell, editors, *Purposive Systems*, pages 15–23. Spartan Books, 1968.
- [7] P. Fritsch. Five Mellow Guys Follow Their Dream: A ‘Tall Ship’ in Brazil. *The Wall Street Journal*, CXLII(35):1, Friday, February 18, 2000.
- [8] Google. Google Home Page. Google, Inc. Mountain View, CA, June 2001.
- [9] J.L. Gould and C.G. Gould. *The Honey Bee*. Scientific American Library, New York, 1988.
- [10] S.G. Lee, T.U. Jung, and T.C. Chung. An Effective Dynamic Weighted Rule for Ant Colony System Optimization. In *Proceedings of CEC 2001*, pages 1393–1397. IEEE, Piscataway, NJ, 2001.
- [11] M. Lindauer. *Communication Among Social Bees*. Harvard University Press, Cambridge, Massachusetts, 1961.
- [12] F. Marckini. *Search Engine Positioning*. Wordware Publishing, Inc., Plano, TX, 2001.
- [13] D. Merkle, M. Middendorf, and H. Schmeck. Ant Colony Optimization for Resource Constrained Project Scheduling. In *Proceedings of GECCO 2000*, pages 893–900. Morgan Kaufman Publishers, Inc., 2000.
- [14] T. Ray and K.M. Liew. A Swarm with an Effective Information Sharing Mechanism for Unconstrained and Constrained Single Objective Optimization Problems. In *Proceedings of CEC 2001*, pages 75–80. IEEE, Piscataway, NJ, 2001.
- [15] A. Vakali. A Web-based Evolutionary Model for Internet Data Caching. In *Proceedings of the 10th International Workshop on Database and Expert Systems Applications*, pages 650–654. IEEE Press, 1999.
- [16] R.L. Walker. Preliminary Study of Web Scouts/Foragers for a Bioinformatic Application: A Parallel Approach. In Y.V. Esteve, G.M. Carlomagno, and C.A. Brebbia, editors, *Computational Methods and Experimental Measurements X*, pages 967–976. WIT Press, 2001.
- [17] R.L. Walker. Applying Evolutionary Computation Methodologies for Search Engine Development. In *SEAL’02: Proceedings of the 2002 Asia-Pacific Conference on Simulated Evolution and Learning*, November 2002. To appear.
- [18] R.L. Walker. Using Nearest Neighbors to Discover Web Page Similarities. In H.R. Arabnia, editor, *PDPTA’02: Proceedings of the 2002 International Conference on Parallel and Distributed Processing Techniques and Applications*, pages 157–163. CSREA Press, June 2002.
- [19] M. Wineberg and F. Oppacher. Enhancing the GA’s Ability to Cope with Dynamic Environments. In *Proceedings of GECCO-2000*, pages 3–10. Morgan Kaufman Publishers, Inc., 2000.

Author Index

Abe, J.M.	775	Fdez-Riverola, F.	32
Abraham, A.	321,853	Ferreira, C.	153
Aguilar, J.L.	426,681	Figueiras-Vidal, A.R.	365
Ait Oufroukh, N.	673	Frydman, C.S.	231
Amali, R.	15	Fu, H.C.	818
Angkawattanawit, N.	573	Garaffa, I.M.	490
Aoki, T.	480	Garitagoitia, J.R.	115
Arenas-García, J.	365	Glen, E.	173
Asseraf, M.	212	Gokcen, I.	280
Astrain, J.J.	115	Gouarderes, G.	529
Autrel, F.	806	Gouarderes, S.	529
Baeza-Yates, R.	565	Gutiérrez, C.	593
Batista, G.E.A.P.A.	251	Hadany, L.	193
Beker, T.	193	Haindl, M.	697
Benferhat, S.	806	Hashimoto, S.	54
Boblan, I.	183	Heinen, F.J.	830
Bologna, G.	42	Hirche, S.	183
Brouwer, R.K.	707	Hruschka, E.R.	453
Buckles, B.P.	280	Hruschka, E.R. (Jr.)	453
C.N. Purdy	735	Iba, H.	540
Cang, S.	394	Inui, T.	64
Carvalho, P.	341	Ishibuchi, H.	132,163
Castilho, V.C.	796	Jackson, W.	84
Castillo, C.	565	Jakovlevich, C.A.	462
Castillo, O.	755	Jarur, M.C.	602
Castro, M.J.	644	Jedrzejowicz, J.	241
Catucci, G.	513	Jedrzejowicz, P.	241
Cavalcante Prudêncio, R.B.	74	Kacprzyk, J.	417
Chen, Y.H.	818	Kawamae, N.	300
Cheng, S.S.	818	Knowles, J.D.	271
Cios, K.J.	84	Kokol, P.	374
Colle, E.	673	Köppen, M.	309,765
Corchado, J.M.	32	Kramer, K.-D.	785
Córdoba, A.	115	Kumar, V.S.	625
Corne, D.W.	271,290,881	Le Goc, M.	231
Cuppens, F.	806	Lee, S.W.	21
Dahlquist, E.	221	Lenič, M.	374
De Cock, M.	142	Letelier, J.C.	205
Delpy, P.	529	Liu, X.	765
Dixon, P.W.	290	Liu, Z.	331
Do Nascimento, H.A.D.	634	Ludermir, T.B.	74
Eades, P.	634	Lyhyaoui, A.	365
Ebecken, N.F.F.	453	Macedo, S.	470
El Debs, M.K.	796	Madsen, A.	221
Fariña, F.	115	Mamdani, E.	470

Marik, V.	550	Sanchis, E.	644
Marín, G.	205	Santibanez-Koref, I.	183
Marín, M.	612	Sepulveda, R.	755
Martins, M.C.	490	Sharma, D.	433
Mashkov, V.	550	Shawkat Ali, A.B.M.	321
Mastropasqua, D.	124	Shekar, B.	95
Maturana, C.	384	Shirai, K.	745
Melin, P.	755	Šimberová, S.	697
Miège, A.	806	Simon, R.	519
Minami, T.	64	Sinka, M.P.	881
Monard, M.C.	251	Smith, K.A.	843,853
Montiel, O.	755	Sood, A.K.	519
Mora-Jiménez, I.	365	Soto-Andrade, J.	205
Morales, E.K.	593	Springhetti, L.	84
Mosca, N.	124	Sugimoto, F.	652
Mpodozis, J.	205	Sunayama, W.	863
Muge, F.	500	Sundareshan, M.K.	261
Mukkamala, S.	351	Sung, A.H.	351
Murakami, M.	745	Suzuki, A.	775
Murty, M.N.	690	Swiercz, W.	84
Nacke, T.	785	Tepper, J.	21
Nakamatsu, K.	775	Tokui, N.	540
Natarajan, R.	95	Torres, S.	715,725
Navarrete, P.	663	Tseng, C.L.	818
Navia-Vázquez, A.	365	Veenhuis, C.	309
Nickolay, B.	765	Velásquez, J.D.	480
Nicoletti, M.C.	796	Vera, E.	725
Nolan, J.J.	519	Vicente Garcia, R.	765
Noroozi, S.	15	Villadangos, J.	115
Oates, M.J.	290	Vinney, J.	15
Osório, F.S.	830	Vishwanthan, S.V.N.	690
Palmer-Brown, D.	21	Wakaki, H.	540
Pao, H.T.	818	Walker, R.L.	891
Partridge, D.	394	Wang, X.	843,853
Patel, V.	15	Weber, R.	384,480
Patzwahl, S.	785	Weidl, G.	221
Peng, J.	280	Wiese, K.C.	173
Perozo, N.	426	Xu, Y.	331
Pezoa, J.	715	Yachida, M.	863
Pina, P.	500	Yamamoto, T.	132
Purdy, C.N.	735	Yasuda, H.	480
Rakus-Andersson, E.	105	Yoneyama, M.	652,745
Ramos, V.	500	Yoshida, T.	163
Reeves, R.	725	Youssif, R.S.	735
Ribeiro, B.	341	Zadrožny, S.	417
Roadknight, C.	21	Zakrzewski, L.	105
Rodríguez Tastets, M.A.	583,602	Zambetta, F.	124,513
Ruiz-Del-Solar, J.	663	Zanni, C.H.	231
Rumantir, G.W.	871	Zegers, P.	261
Rungsawang, A.	573	Zemke, S.	404
Saegusa, R.	54	Ziarko, W.	442